

AD-A248 776

REF ID: A248 776

UNITED STATES AIR FORCE

SUMMER RESEARCH PROGRAM -- 1991

**HIGH SCHOOL APPRENTICESHIP PROGRAM
(HSAP) REPORTS**

VOLUME 13

WRIGHT LABORATORY



RESEARCH & DEVELOPMENT LABORATORIES

5800 UPLANDER WAY

CULVER CITY, CA 90230-6608

SUBMITTED TO:

**LT. COL. CLAUDE CAVENDER
PROGRAM MANAGER**

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

BOLLING AIR FORCE BASE

WASHINGTON, D.C.

DECEMBER 1991



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 9 January 1992	3. REPORT TYPE AND DATES COVERED 30 Sep 90-30 Sep 91	
4. TITLE AND SUBTITLE 1991 High School Apprenticeship Program (HSAP) Volumes 10-13 Vol. 13			5. FUNDING NUMBERS F49620-90-C-0076	
6. AUTHOR(S) Mr Gary Moore				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Research Development Laboratories (RDL) 5800 Uplander Way Culver City CA 90230-6608			8. PERFORMING ORGANIZATION REPORT NUMBER AFOSR-TR- 92 0180	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFOSR/NI Bldg 410 Bolling AFB DC 20332-6448 Lt Col V. Claude Cavender			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT UNLIMITED			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) High school students who live in communities where Air Force laboratories are located have an opportunity to spend eight weeks during the summer doing scientific research at the laboratory. Each student is assigned a mentor from the laboratory. During the summer of 1991 132 students participated in the program. Each student was required to submit a report on their accomplishments. Those student reports were consolidated and bound into this annual report.				
14. SUBJECT TERMS			15. NUMBER OF PAGES	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

UNITED STATES AIR FORCE
SUMMER RESEARCH PROGRAM -- 1991
HIGH SCHOOL APPRENTICESHIP PROGRAM (HSAP) REPORTS

VOLUME 13
WRIGHT LABORATORY

RESEARCH & DEVELOPMENT LABORATORIES
5800 Uplander Way
Culver City, CA 90230-6608

Program Director, RDL
Gary Moore

Program Manager, AFOSR
Lt. Col. Claude Cavender

Program Manager, RDL
Claude Baum

Program Administrator, RDL
Gwendolyn Smith

Submitted to:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

Bolling Air Force Base

Washington, D.C.

December 1991

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Availability for Special
A-1	



PREFACE

Reports in this document are numbered consecutively beginning with number 1. Each report is paginated with the report number followed by consecutive page numbers, e.g., 1-1, 1-2, 1-3; 2-1, 2-2, 2-3.

This document is one of a set of 13 volumes describing the 1991 AFOSR Summer Research Program. The following volumes comprise the set:

<u>VOLUME</u>	<u>TITLE</u>
1	Program Management Report
	<i>Summer Faculty Research Program (SFRP) Reports</i>
2	Armstrong Laboratory, Wilford Hall Medical Center
3	Phillips Laboratory, Civil Engineering Laboratory
4	Rome Laboratory, Arnold Engineering Development Center, Frank J. Seiler Research Laboratory
5	Wright Laboratory
	<i>Graduate Student Research Program (GSRP) Reports</i>
6	Armstrong Laboratory, Wilford Hall Medical Center
7	Phillips Laboratory, Civil Engineering Laboratory
8	Rome Laboratory, Arnold Engineering Development Center, Frank J. Seiler Research Laboratory
9	Wright Laboratory
	<i>High School Apprenticeship Program (HSAP) Reports</i>
10	Armstrong Laboratory
11	Phillips Laboratory, Civil Engineering Laboratory
12	Rome Laboratory, Arnold Engineering Development Center
13	Wright Laboratory

1991 HIGH SCHOOL RESEARCH REPORTS

Wright Laboratory

<u>Report Number</u>	<u>Report Title</u>	<u>Author</u>
Aero Propulsion & Power Laboratory (PROP)		
1	Equal Distribution of Pressure in a Facility Containing Unequal Flow Velocities	Lee Bunch
2	The Producing and Testing of Single Cell Thermal Batteries	Tina Chilcoat
3	A Study in Solid and Liquid Lubricants	Jeffrey Cropp
4	Fuel	Matthew Grider
5	Static Wicking	Mark Hansell
6	Final Report, Summer 1991	Christopher O'Dell
7	Compressor Research Facility Test Group	Jennifer Pollock
Armament Laboratory (ATL)		
8	Space Based Interceptor Star Tracker Analysis	Eric Apfel
9	Preparation and Characterization of 3-Picrylamino- 1,2,4-Triazole	Kathryn Deibler
10	Mercury-Indium-Silver Alloys for Joining Copper and Aluminum Striplines	Lesha Denega
11	Using the IEEE-488 General Purpose Interface Bus for Device Integration	Brian Eplett
12	Hydrocode Animation	Tom Fraites, Jr.
13	Stress Sensor Experimentation, Analysis and Evaluation	Daniel Grayson
14	Transforms, Image Compression	Derek Holland
15	Visible Laser Polarimetry	Chad Houghton
16	Measurements to Determine Mechanical Characteristics of Materials	Jason Kitchen
17	Neural Network Implementation of an Extended Kalman Filter	Daran Mason
18	Quick-Look, Video-Based Scoring for Captive Flight Testing of ASARG	Christine Riendeau
19	Low-Cost Coherent-On Receive Missile Radar	Dennis Scott, Jr.
20	Digital Signal Processing	Troy Urquhart
21	Interior Ballistics Applications in Aerospace Research	Greg VanWiggeren

<u>Report Number</u>	<u>Report Title</u>	<u>Author</u>
<u>Wright Laboratory (cont.)</u>		
22	Randomly Generated Synthetic Background Data Using Iterative Fractal Techniques	Eric White
23	Software Development for the Aerosol Test Chamber and a Code Conversion for a Data Retrieval System	James Youngblood
Avionics Laboratory (AVION)		
24	Final Report	Lori Anderson
25	Communication, Navigation, Identification Laboratory Research	Barry Koestler
26	The Use of Prolog and Rule Based Systems in Artificial Intelligence	Allen Lefkowitz
27	Summary of Research on a new Radar Scenario	Meredith Lewis
28	Radar Study	Eric Powers
29	Communication, Navigation, Identification Laboratory Research	Karen Thomas
30	PT 2Memo: Chaotic Functions Experiment	Christine Yoon
Electronic Technology Laboratory (ETL)		
31	Monte Carlo Simulation of 50-keV Electrons in Various Resists	Mark Buxton
32	Final Report	Helen Chou
33	What I Did for My Summer Vacation	Alan Page
34	The Third-Order Intercept Point in Microwave Circuits	Suzette Yu
Flight Dynamics Laboratory (FDL)		
35	Final Report	Jean Ay
36	Contract with the Computer Resources Team	Matthew Becker
37	Saving Time, Money, and Lives with CFD	Shari Goldenberg
38	WPAFB FIBT Structural Testing: HSAP 1991	Christina Ho
39	Report of a Summer	Robert Horrocks, Jr.
40	My Summer at The Aircraft Survivability Research Facility	Cathie Moore
41	A Summer at FIBT	Eugene Paige
42	HSAP Final Report	Michael Richie

EQUAL DISTRIBUTION OF PRESSURE IN A FACILITY CONTAINING UNEQUAL FLOW VELOCITIES

Tecumseh High School Senior Student, Lee Bunch

Abstract

In gaseous two-dimensional shear-layer flows from a splitter plate, it is observed in a wind chimney containing unequal flow velocities that the shear-layer tends to slant toward the low velocity side. This occurs due to unlike pressure distributions along the low velocity (right) and high velocity (left) walls of the containment chimney. A pivotal wall on the low velocity side of the chimney is proposed as a solution to the pressure distribution problem. Pressure taps are located in the left and right walls in order to help in positioning the right wall to where the pressure distribution is equal on both sides of the splitter plate, and the vortices formed do not collide with the chimney walls. Based on previous experiments and data, it is predicted that the procedure mentioned keeps the shear-layer from slanting to the low pressure side of the flow and equalizes the influence of the walls on the vortices formed.

1. Introduction

Combustion is a complex process which can be broken up into several less complex processes for experimentation and indepth

research. An essential process in the research of combustion is the study of fluid dynamics and droplet/vortex interactions. Lt. Robert Hancock of the Aero Propulsion and Power Laboratory at Wright-Patterson Air Force Base has performed a variety of experiments involving gaseous flow and found excellent information to further the understanding of combustion. In his experiments, a variety of characteristics from the gaseous flow are examined using advanced diagnostic techniques and equipments.

Lt. Hancock uses a fairly simple experimental design, but it proves to be truly unique. In the experiment, vortices are formed from a splitter plate which separates a high and low velocity air flow contained within a square Plexiglass wind chimney. Seed is then introduced into the flow by mixing very small Phenolic microballons into the air flow. By strobing the particles with first a blue and then a green laser sheet, the distance and direction of the particles can be measured of a photographic image. Another more advanced and unique method is accomplished by passing the air on the high speed side of the splitter plate over a liquid TiCl_4 bath in order to collect TiCl_4 vapor. Once the TiCl_4 has entered the wind chimney, it reacts with water droplets injected through a slot in the Plexiglass wind chimney from the low speed dry air flow. The water droplets are formed using a droplet generator driven by a piezo electric crystal transducer. Upon crossing the shear-layer into the high-speed, TiCl_4 -laden side of the flow, water vapor leaving an evaporating droplet reacts with the TiCl_4 vapor forming micron-

sized TiO_2 seed particles at the interface of the two air flows. These particles follow the gas flow and distinctly mark the convective transport of the water from the droplet. Green light from a doubled frequency Nd:Yag laser is used to illuminate the seed particles, allowing one to simultaneously observe both the influence of the gaseous flow of the droplet, and the path that the vapor which originates from the droplet follows. In order to help one visualize the experiment a schematic has been included with this report (figure1).

Although the experimental design used in this current observation of air flow is well developed, a natural problem exists in the design of the Plexiglass wind chimney. In the wind chimney the splitter plate divides the chamber into two equal areas once the chimney has been fastened in position over the air flow. Since the areas on both sides of the splitter plate are equal, and the velocities of the flows on both sides of the splitter plate are unequal, this causes the pressure on the high velocity side to be greater than the pressure on the low velocity side. This unequal pressure distribution causes the shear-layer and the vortices formed to slant toward the low velocity side. The slanting of the shear-layer and the vortices formed causes an interference in the flow once the vortices begin to interact with the wall (figure 2). At the maximum velocities of each flow, which are 1.3 m/s for the high velocity side and 0.5 m/s for the low velocity side, the walls interfere with the

flow closer to the splitter plate, making the collection of accurate information almost impossible.

The solution reached through research and observation simply involves construction of a new "variable wall" in the Plexiglass wind chimney. By modifying the design of the chimney so that the low velocity side wall does not interfere with the flow, a more controlled environment can be achieved for experimentation and data collection. In order to prevent the wall on the low velocity side of the flow from interfering with the flow, a pivotal wall is proposed along with various pressure taps on both the high and low velocity side walls. By placing pressure taps along the high and low velocity side walls parallel to each other, one can position the low velocity wall to make sure that the pressure is distributed equally along both walls, and that the shear-layer does not slant toward the low velocity side of the flow.

The final experimental results of the Plexiglass wind chimney have not been documented in this report because of the deadline of the research paper, and also because of the length of the construction time. The results achieved for documentation are the final design of the wind chimney along with its completed construction. Several photographs (figure 3) of the recently constructed chimney are included. Experimental testing for final results of the Plexiglass wind chimney will be performed by Lt. Robert Hancock at a later date.

2. Discussion

Upon observing the experiment involving two different velocity air flows forming vortices over a splitter plate which divides the wind chimney into two equal areas, the slanting of the shear-layer due to unequal pressure distribution is clearly observed (figure 4), along with other conditions which are essential to the experiment and must be considered. Since the gaseous flow is observed in this experiment, it is best not to interfere with the gaseous shear-layer flow. Therefore the low velocity wall, which the turbulent air flow containing the vortices slants into, must be moved so as not to interfere with the flow. All other surfaces exposed to the air flow inside the air chamber also must not interfere with the air flow. It is also imperative to keep the air flow which is discharged from the air jet contained within a sealed chamber with optical access. The sealed air chamber is necessary in order to prevent air currents in the test area from interfering with the air flow. Optical access is essential in order to bring the lasers into the flow field so as to perform laser diagnostics. The splitter plate is permanently positioned in the air flow in such a way that it separates the outlets of the high and low velocity air flows equally. Any type of air chamber obtained for the experiment must be able to be positioned on top of the air jet with a foundation that can be bolted down by ten 9/32 inch holes. It must also have a small slot centered in the low velocity wall near the splitter plate so as to maintain access for the droplet generator.

The main problem existing in the experiment still remains in the slanting of the shear-layer and the vortices formed from the splitter plate. The shear-layer and the vortices formed are fundamental in gaseous flow. When two unequal velocity air flows combine a shear-layer is created. The shear-layer contains unstable waves which roll into vortices. The slanting of the shear-layer and the vortices formed is caused essentially from a high velocity air flow being maintained on one side of the turbulent flow while a low velocity air flow is being maintained on the other side. Combined with the current equal areas maintained on both sides of the splitter plate, the pressure on the high velocity side of the air flow pushes the shear-layer air flow towards the low velocity side of the flow. Finally, when the vortices reach the low velocity wall interference with the flow results and the experiment becomes more complex. It becomes clear that the construction of a modified Plexiglass wind chimney, that can accommodate the slanting shear-layer, is essential to the collection of accurate data.

Through in depth research and various discussions with engineers, a solution involving an improved design of the Plexiglass wind chimney was completed in order that it may function properly under the conditions required. Upon investigating a variety of similar experiments, it was learned through an experiment involving density effects and large structures in turbulent mixing layers, performed by Garry Brown and Anatol Roshko, that by using a pivotal wall along with pressure taps one can keep the mixing layer parallel

to the splitter plate. At this point it is clear that a pivotal wall on the low velocity side of the flow is the best method to prevent the wall from interfering with the slanting shear-layer and the vortices formed above the splitter plate. In order to guide the low velocity wall into the correct position, pressure taps are placed in the low and high velocity walls parallel to each other. The pressure taps measure the static pressure at various positions on the walls of the chimney. The measurements from the pressure taps enable one to maintain equal pressure distribution on both the high and low velocity walls. It is this equal distribution of pressure which keeps the mixing layer parallel to the splitter plate. Through the information gathered, it is clear that a modified Plexiglass wind chimney, having a low velocity pivotal wall and pressure taps capable of measuring the pressure distributed on the high and low velocity walls, will enable one to perform the experiment involving the observation of gaseous flow in a facility containing unequal flow velocities.

3. Results

The final design of the Plexiglass wind chimney is much more complicated than the original chimney in that it prevents the wall on the low velocity side of the flow from interfering with the slanting of the shear-layer and the vortices formed. Maintaining the conditions required for experimentation, the new chimney allows the experimental flows to be run at higher velocities. The modified

Plexiglass wind chimney is designed with newly altered parts. It is made up of a foundation, a pivotal low velocity side wall connected to the foundation by a rubber hinge, a permanent high velocity side wall, and two permanent front and rear walls. These changes make the chimney more effective for the study of gaseous shear-layer flows.

The modified foundation is designed so that it acts as a pivot point for the movable low velocity wall, and so that it maintains a tighter seal with the other three walls, making the walls more secure and stable. The original foundation supports the walls by simply having them glued on top of it, unlike the modified foundation. In the new model special slots are added to the foundation, which is thickened in order to accommodate the slots and add extra strength. By placing slots along the inner portion of the high velocity, front, and rear sides of the new foundation, these walls are exposed to a greater amount of surface area of the foundation. The larger amount of exposed area provides the glue with more surface area to seal, making a tighter and more efficient seal. The slots also give extra support to the walls, which is greatly needed. In order to provide a pivot point for the low velocity wall no slot was made in the side of the foundation, instead slots were made extending through the bottoms of the front and rear sections of the foundation. These slots accommodate the extensions of the rubber hinge which are attached to the low velocity wall. This enables the low velocity wall to be secured to the foundation,

but by using a rubber hinge the low velocity wall can still pivot left and right. Although the foundation is greatly modified it still has ten 9/32 inch holes which allow it to be secured to the splitter plate assembly. The modified foundation also maintains the correct dimensions for it's inner opening so that the flow enters the wind chimney undisturbed. The foundation is clearly instrumental in maintaining the wind chimney's structure.

The low velocity pivotal wall, which is located on the right side of the wind chimney, is the most vital element of the facility, along with it's attached rubber hinge and pressure taps. The right wall is made thicker than the other walls in order to accommodate the rubber hinge and rubber seals. The low velocity wall is designed so that it is only attached to the foundation by the rubber hinge. Connected only to the inner portion of the wall, the rubber hinge is used as a smooth transitional surface from the foundation to the low velocity wall, when the wall is in a slanted position. The only method applicable to attach the hinge to the low velocity side wall is to screw the rubber directly to the wall. The rubber is positioned in a cut away portion of the wall sized just for the hinge. The remaining section of the wall is beveled at a 15 degree angle in order for the wall to pivot to the right side. Once in position, the rubber is attached to the inner side of the wall by three screws. The screw heads do cause flow interference, but because of the small size of the heads of the screws the interference is minor. Just above the attached rubber, the original slot is maintained so that the

droplet generator may still have access to the flow inside of the wind chimney. Along with the cut away portion for the rubber hinge, a long narrow slot is cut all the way down each side of the wall in order to glue thin strips of rubber in these positions. The rubber strips attached to the sides of the wall act as seals around the wall, since the right wall is not glued to the other walls. Another important characteristic of the right wall is that since it sets on top of the foundation, unlike the other three walls, it extends higher than the other walls. This is done intentionally in order that the right wall may still be at the same height as the other walls when it is slanted. The final modified characteristics found with the low velocity wall are the pressure taps. The pressure taps are positioned left of center so as not to interfere with the slot made for the droplet generator. Each of the four holes drilled for a pressure tap is located at a different height, they are all collinear. Since the walls are not thick enough to accommodate the pressure taps, rounded bosses are glued to the wall centered on each hole. Each boss is carefully sealed by a special glue called dichloromethane, and secured by caulking around the edges for extra seal. It is important to maintain a tight seal in order to get an accurate pressure measurement from the pressure taps. Clearly it is seen that the low velocity wall is complex and plays the key part in controlling the slanting of the shear-layer.

Along with the other altered parts of the Plexiglass wind chimney the front and rear walls are also modified in order to make

the chimney more efficient. The most noticeable modification of the walls are that they extend over three inches beyond the 90 degree position of the pivotal low velocity wall. This is done so as to maintain an inclosed facility around the air flow as the right wall moves out. Another important function of the front and rear walls is to hold the low velocity wall in place since it is only attached to the foundation. Three holes are located in each wall in order to accommodate three tightening rods which clamp the walls tightly around the low velocity wall. This secures the right wall in place, once it is in the correct position, and seals the walls tightly together along with the rubber strips located on the sides of the right wall. The rods can also be loosened in order to reduce stress on the front and rear walls as the right wall is moved to different positions. The holes are positioned angling away from the joint of the right wall so as not to interfere with it's movement. Each wall's equivalent holes are aligned in order to hold the tightening rods properly. The next modified characteristic of each wall is a lower portion which extends beyond the total height. The extended portion of is placed in a cut out section of the foundation in order to maintain a tighter seal with the foundation, when the front and rear walls are tightened around the right pivotal wall. The walls are also positioned in order to maintain a smooth transitional surface from the foundation to the walls. The final characteristic is found only on the rear wall where degree measurements are posted above the low velocity wall. These measurements are used to record data on

the position of the right wall. The front and rear wall modifications are obviously not as vital as that of the low velocity wall, but they are clearly still important.

The high velocity side wall, which is located on the left side of the wind chimney, is only slightly modified. First of all, the entire bottom portion of the wall is placed in a cut out section of the foundation, similar to the front and rear walls. The high velocity wall is glued in place, along with the ends of the front and rear walls located on either side of it, in order to maintain a tight seal and smooth transitional surface from the foundation to the wall. The only other modification of the left wall is insertion of pressure taps. The four taps are drilled right of center, are collinear all the way to the top, and aligned with each equivalent hole on the low velocity wall. The high velocity wall also has round bosses centered and glued over each pressure tap hole exactly like the low velocity wall bosses. These pressure taps are used to measure and match the static pressure profiles along the two side walls. Clearly the left wall is quite simple, but necessary for success.

Upon assembly the Plexiglass wind chimney is sturdy and quite different from the original chimney, while at the same time it still maintains the conditions required for experimentation. The modified wind chimney is mainly constructed out of Plexiglass, except for the rubber strips, hinge, and the screws used to secure the hinge to the right wall. This Plexiglass construction helps the new chimney to

sustain it's optical access for the lasers. The modified low velocity pivotal wall is designed to eliminate the problem flow interference from the slanting shear-layer. Along with the use of pressure taps, equal distribution can be achieved on the walls of the wind chimney. In order to obtain a better understanding of the modified design of the wind chimney, some photographs and copies of the sketches, which include dimensions and assembly instructions, have been included with this report. The modified design of the Plexiglass wind chimney with the pivotal right wall enables one to observe and study gaseous flow more efficiently.

The construction process took delicate work and unfortunately much time. Upon receiving two different estimates on the approximate cost and time needed for construction, Dayton Plastics Inc. was selected to build the chimney. The total cost was \$353.50 with a construction deadline of two weeks, which was exceeded by one week. After receiving the assembled Plexiglass portion of the chimney with right wall separate, the rubber strips and hinge were constructed and attached to the right wall. The right wall was then attached to the rest of the chimney, finalizing assembly at Wright Patterson Air Force Base. As mentioned previously, the experimental results using the Plexiglass wind chimney will be accomplished at a later date by Lt. Robert Hancock.

4. Conclusion

The objective of this work was to equalize and limit the interference of the chimney walls on a two-dimensional shear-layer flow. The shear-layer tended to slant towards the low velocity side wall due to an unequal distribution of pressure along the two walls. The solution reached called for the design and construction of a new Plexiglass wind chimney with a pivotal low velocity side wall and pressure taps. The pressure taps allow the pressure distribution at the two side walls to be monitored and matched so that the influence of the chimney is minimized. This new chimney will allow more controllable and higher speed flows to be studied than previously possible. Although the modified wind chimney did reach final construction, experimental test results could not be achieved for a determination of the chimney's success.

Acknowledgements

Thanks go to Dr W.M. Roquemoire, R.P. Bradley, Dale T. Shouse, and Cindy Obringer for technical assistance and advice in conducting this research. Special thanks go to Lt. Robert Hancock and Frank Timko for extra assistance and advice in the construction and design of the wind chimney.

Figure 1

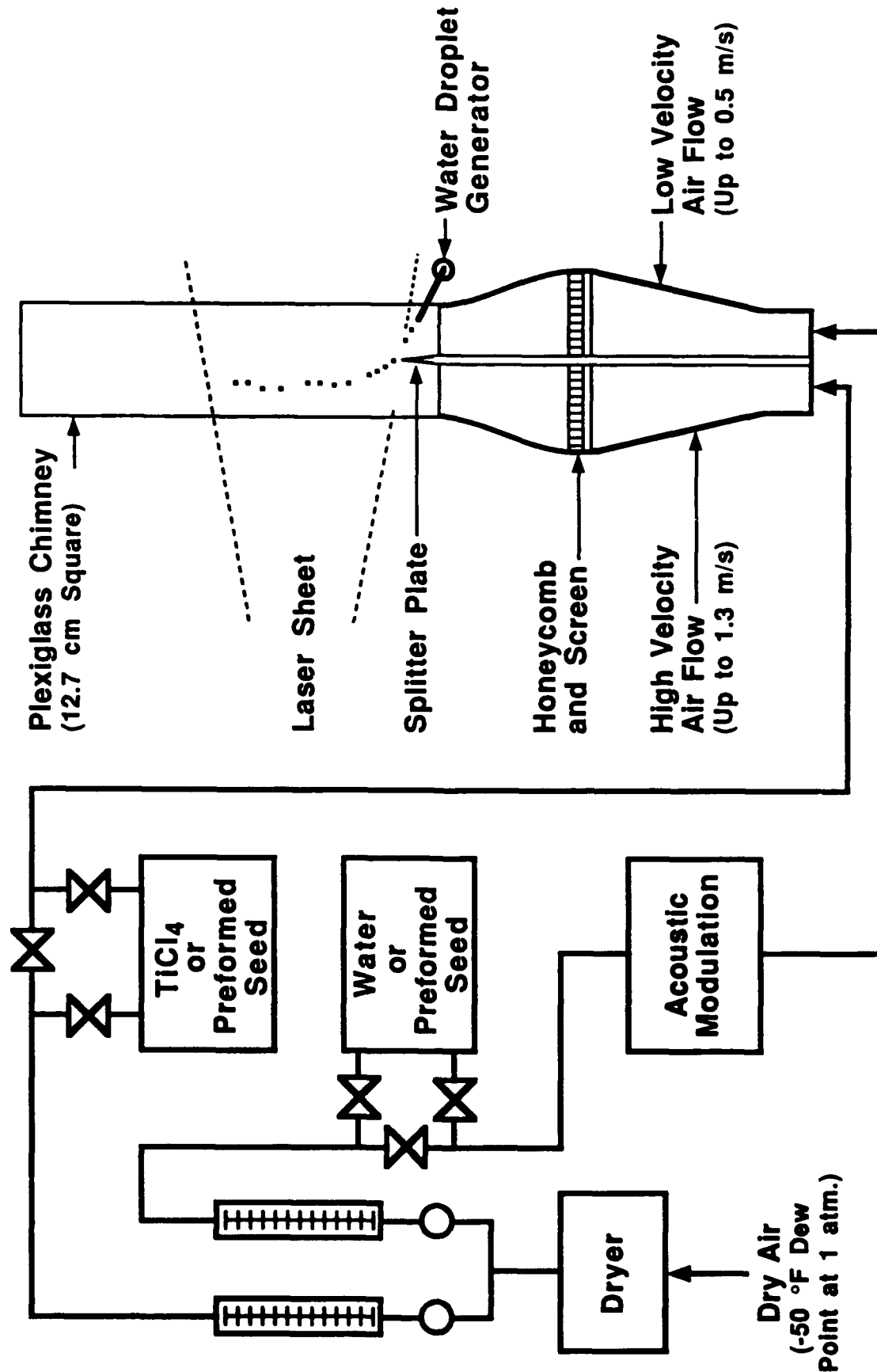


Figure 2

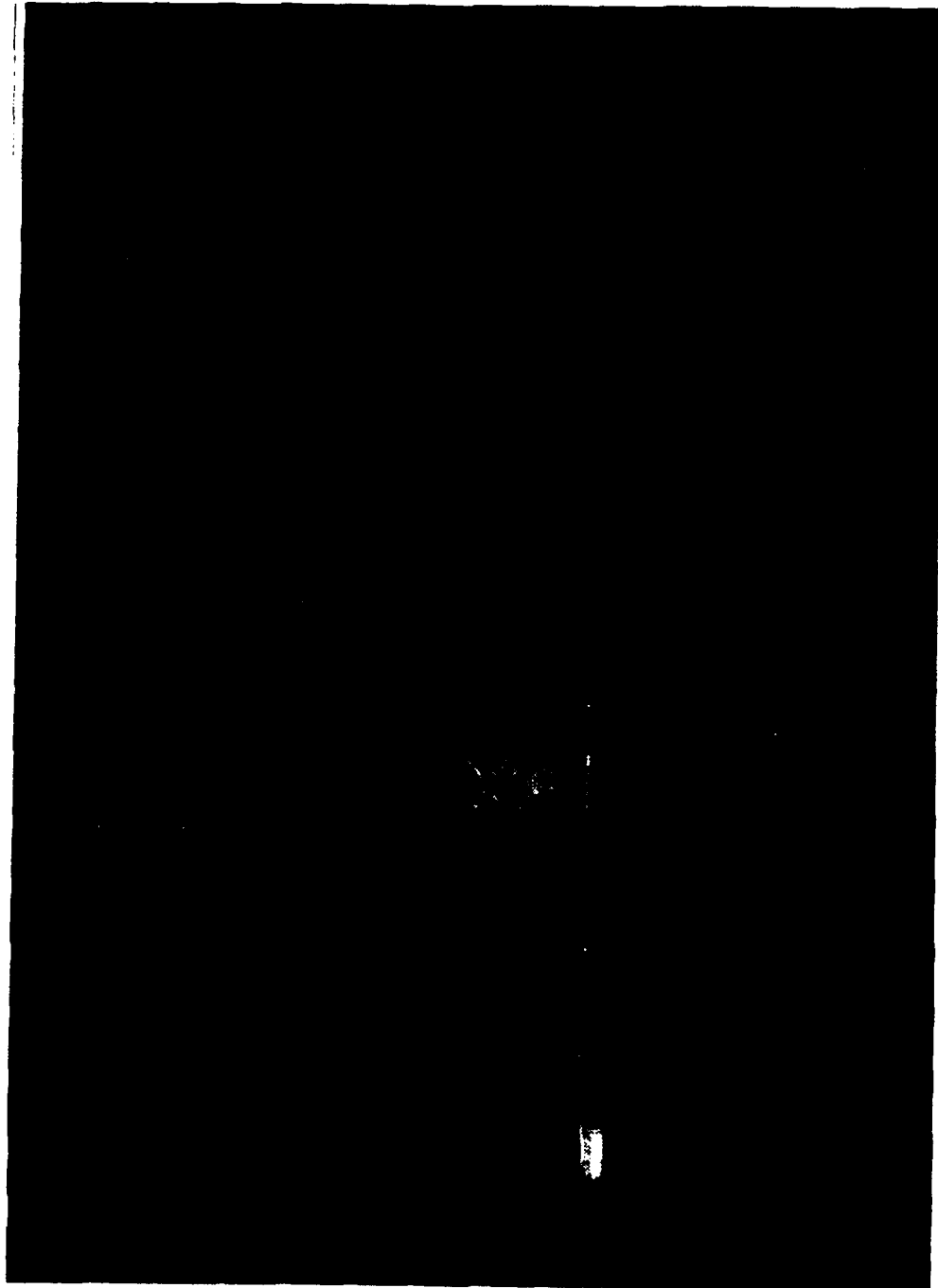


Figure 3

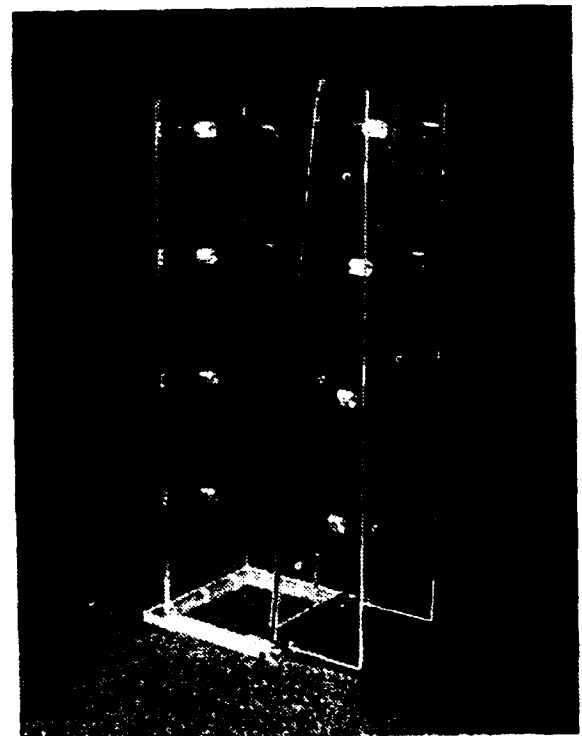
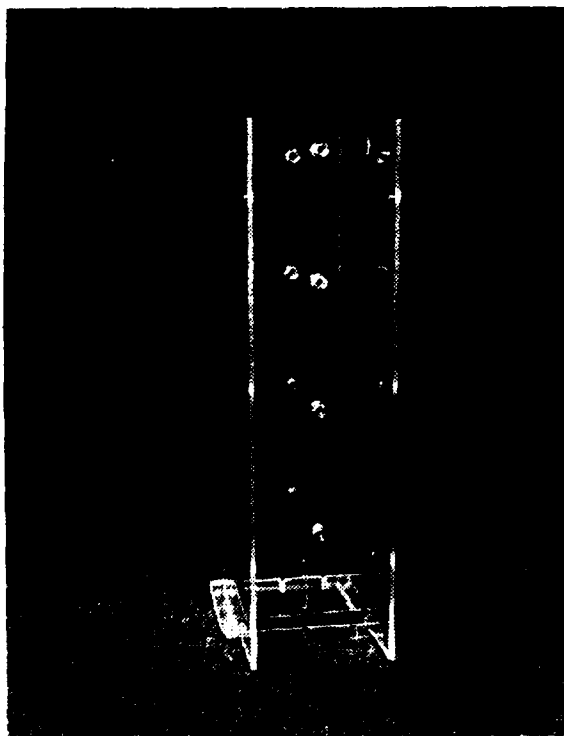
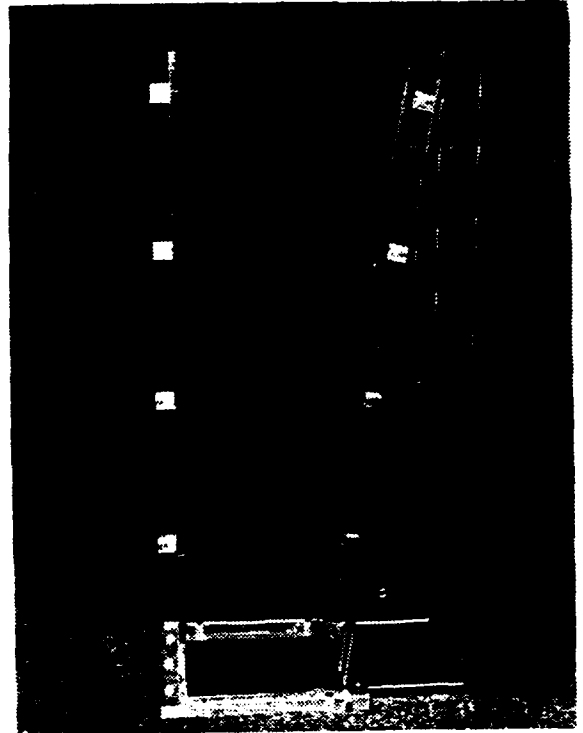
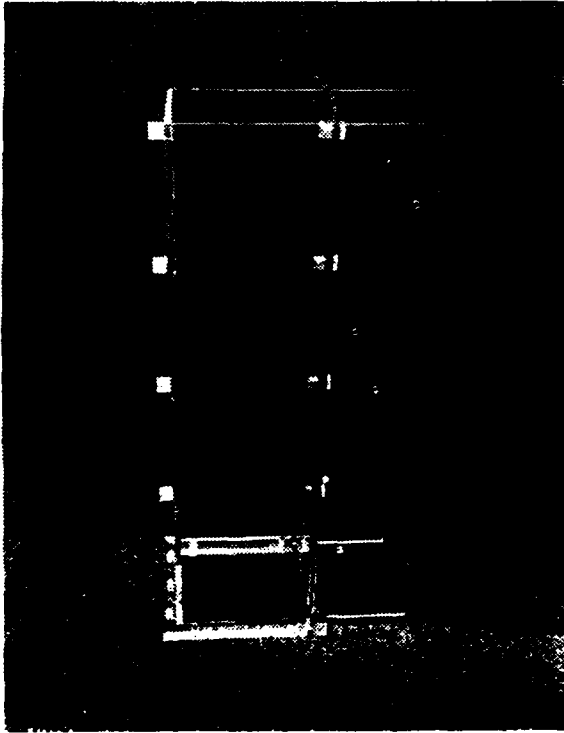
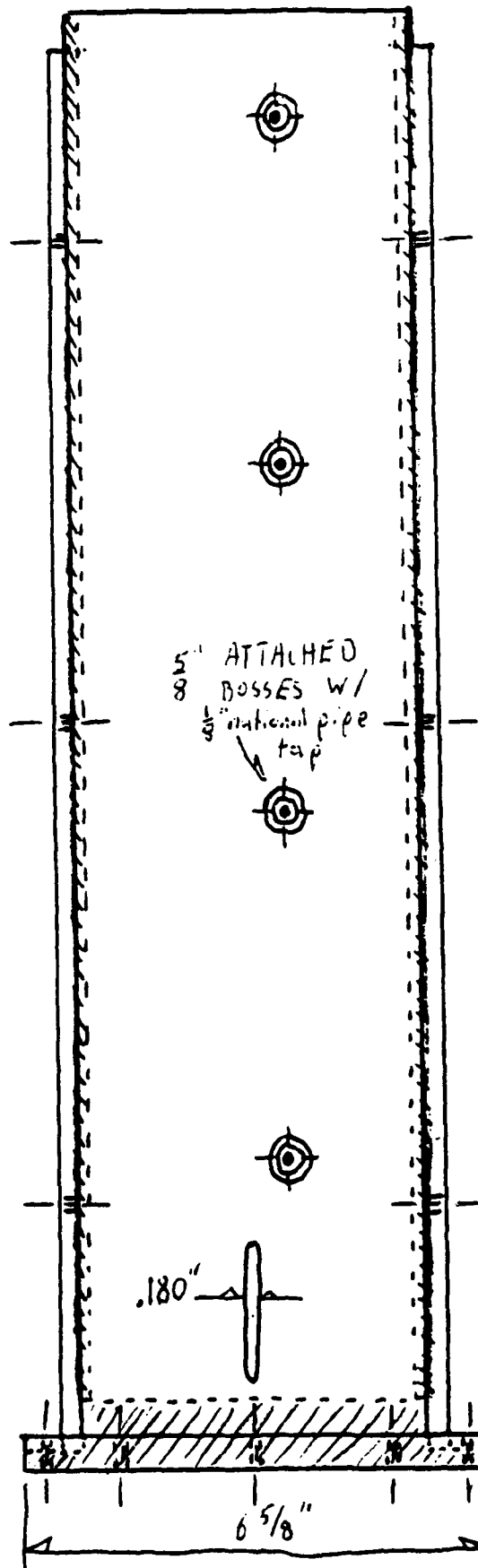


Figure 4



SKETCH: WIND CHIMNEY W/ PIVOTAL RIGHT WALL (RIGHT SIDE VIEW)

-any questions ; call Lee Bunch , 255-6250



* Dimensions located on separate individual sketches

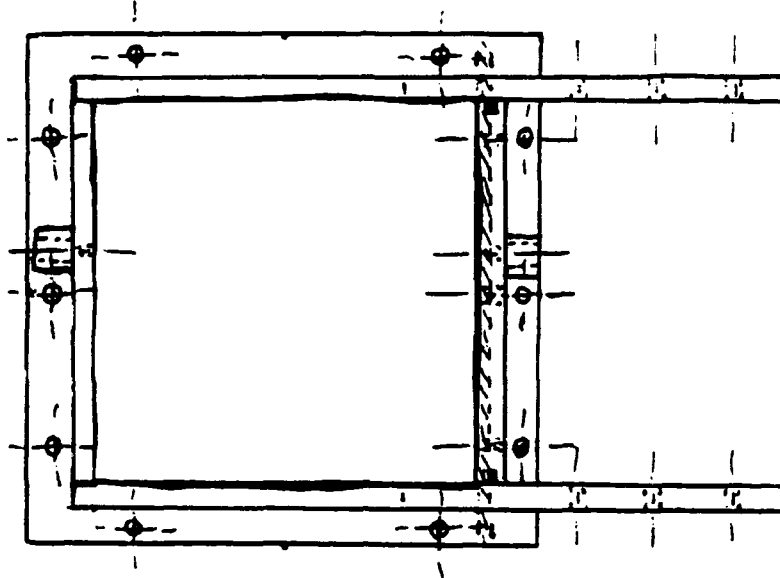
SCALE: $\frac{2}{5}'' = 1''$

1-19

● = RUBBER (exposed)
 ▣ = RUBBER (hidden)

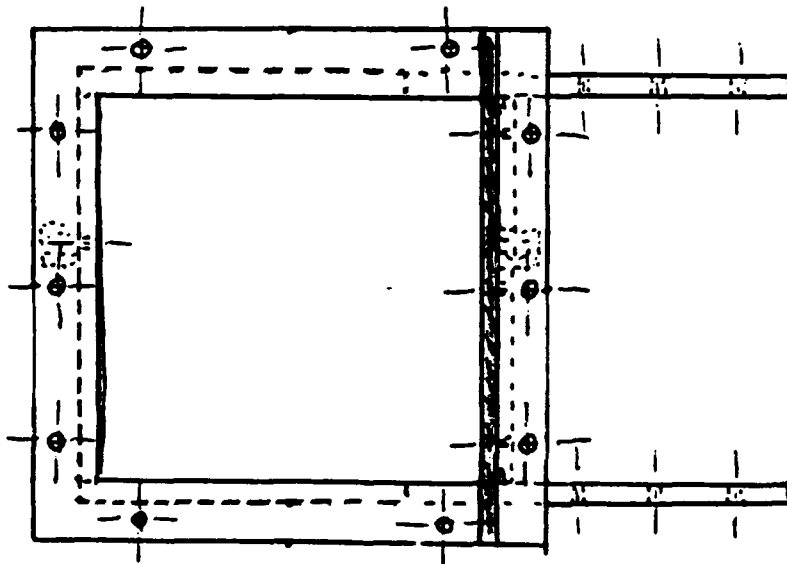
SKETCH: WIND CHIMNEY W/ PIVOTAL RIGHT WALL (TOP & BOTTOM VIEWS)

-any questions; call Lee Bunch, 255-6250



TOP
VIEW

* Dimensions located on
separate individual
sketches



BOTTOM
VIEW

SCALE: $\frac{2}{5}$ " = 1"

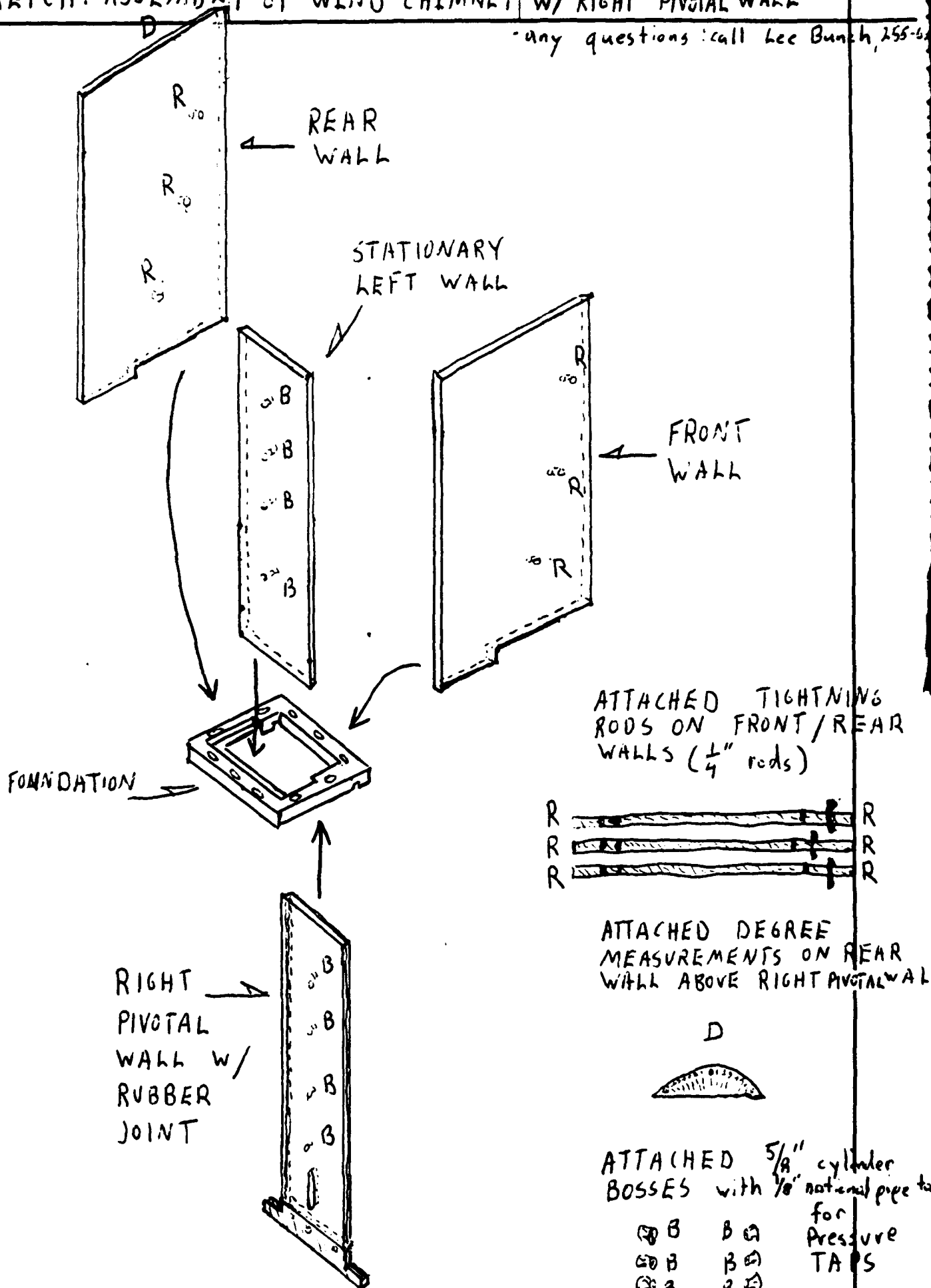
1-20

■ = RUBBER
(exposed)
▨ = RUBBER

SKETCH: ASSEMBLY OF WIND CHIMNEY W/ RIGHT PIVOTAL WALL

any questions call Lee Bunch, 255-6

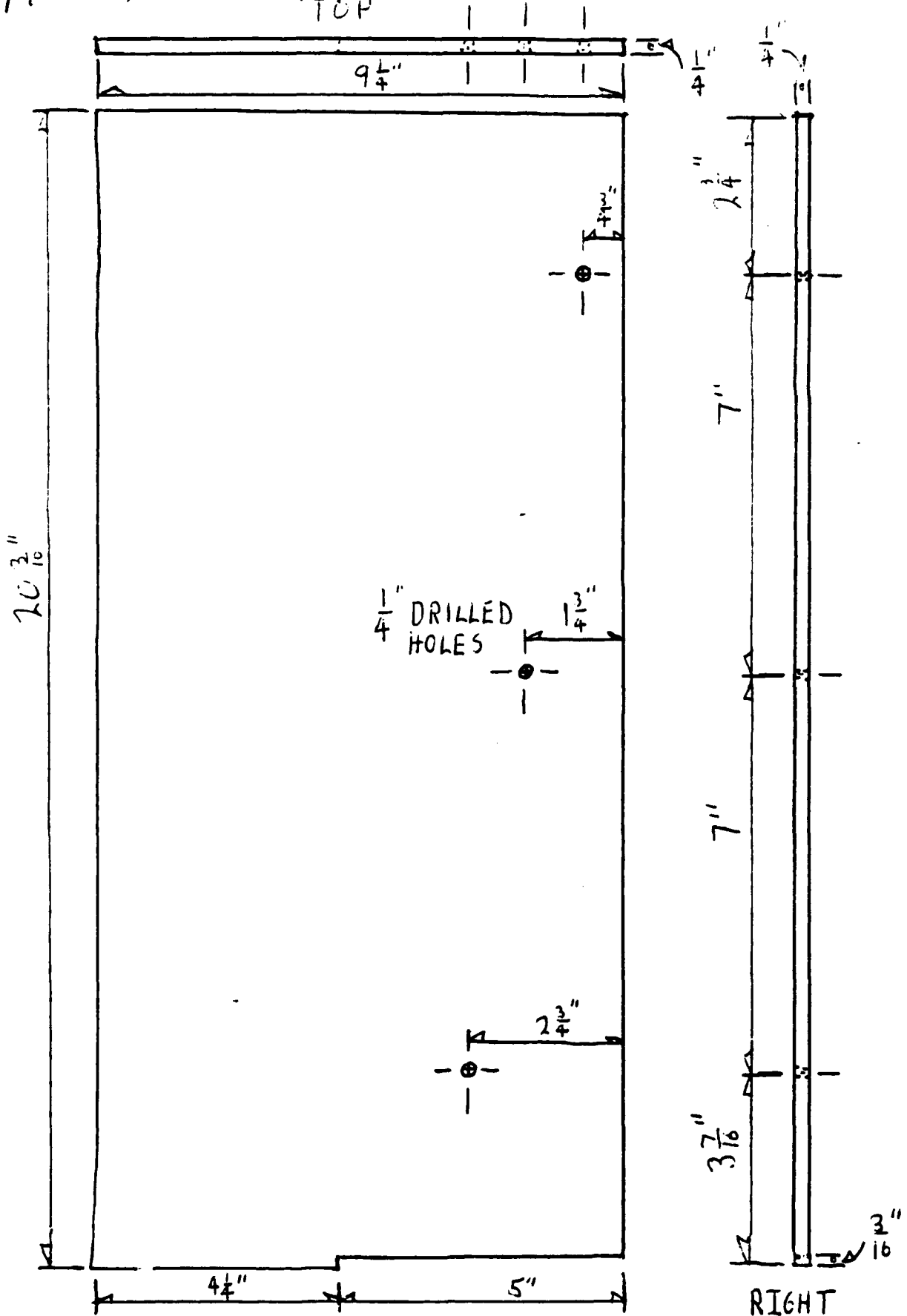
IPC: PNP011
Made in U.S.A



APPROXIMATE SCALE: $\frac{1}{10}$ " = 1" 1-21
(Not Accurate)

SKETCH: FRONT / REAR WALL OF WIND CHIMNEY

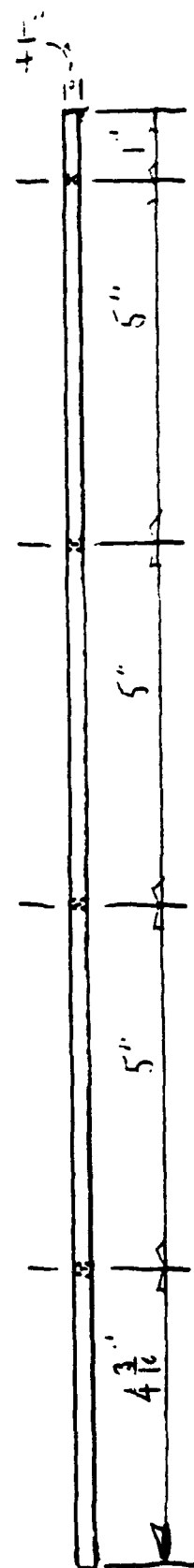
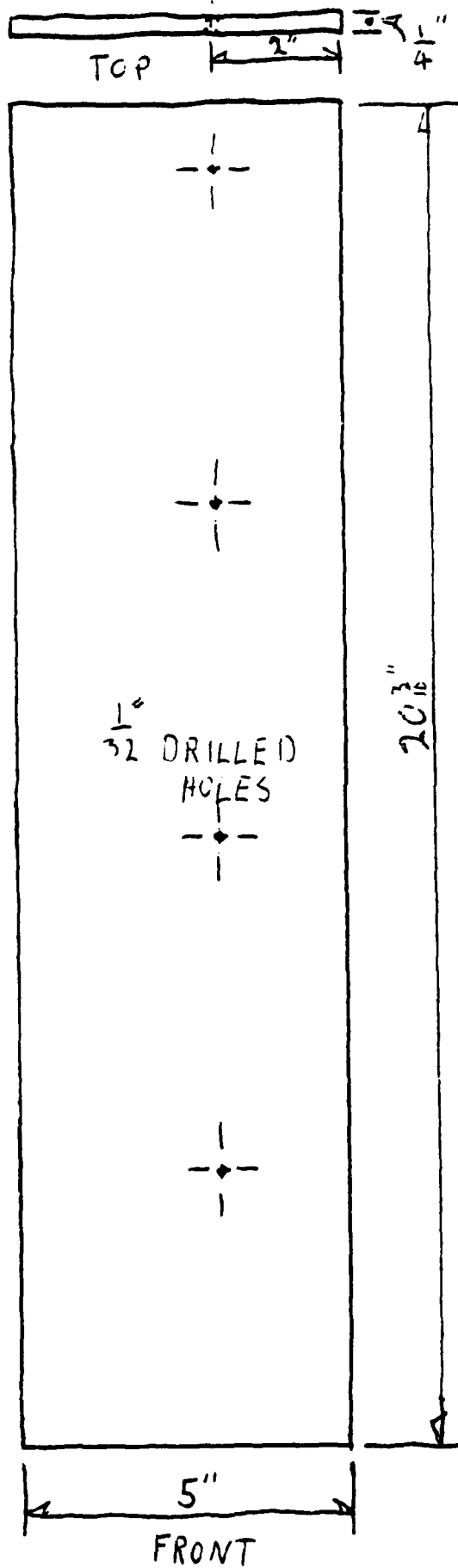
-any questions ; call Lee Bunch, 255-6250
TOP



SCALE : $\frac{2}{5}$ " = 1"

SKETCH: LEFT WALL OF WIND CHIMNEY

-any questions; call Lee Bunch, 255-6250

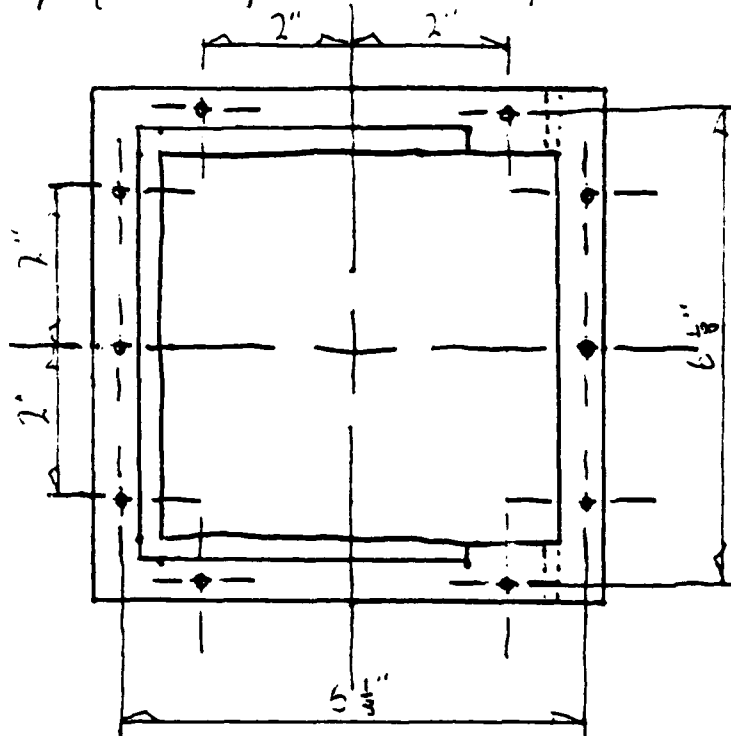


RIGHT

SCALE: $\frac{2}{5}" = 1"$

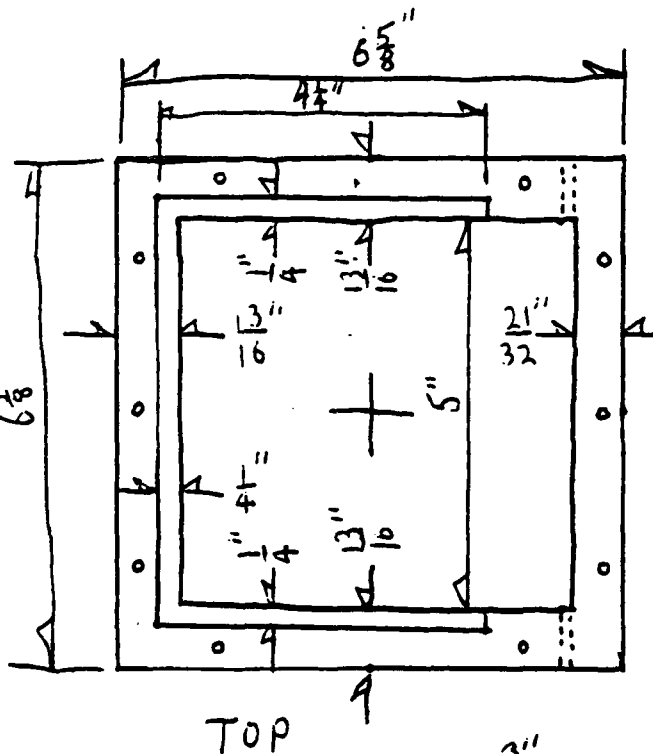
SKETCH: FOUNDATION OF WIND CHIMNEY

any questions; call Lee Bunch, 255-6250



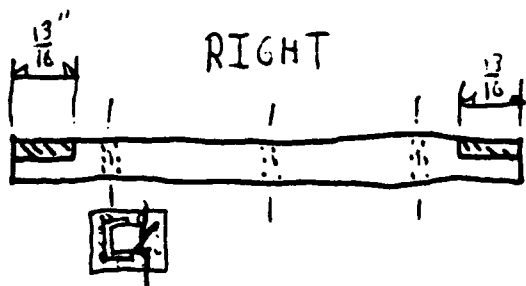
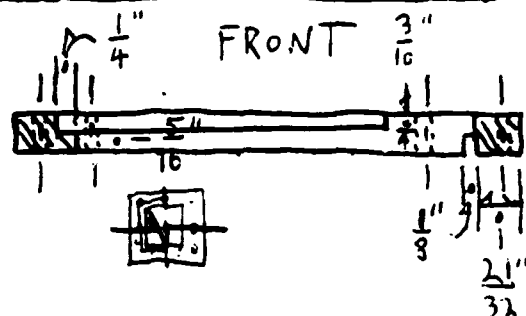
TOP VIEW
WITH
DRILL HOLE
DIMENSIONS

DRILL HOLES = $\frac{9}{32}"$ holes

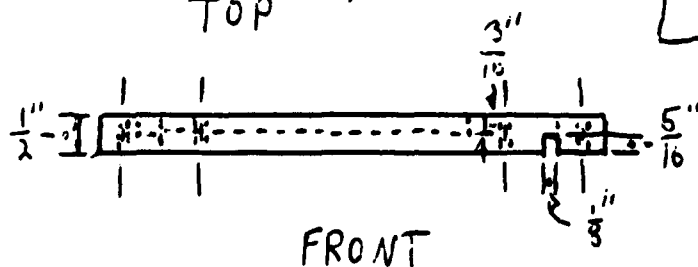


TOP

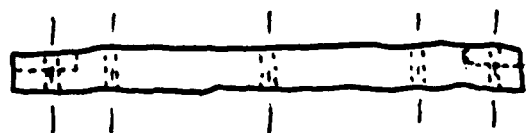
CROSS SECTION VIEWS



RIGHT



FRONT

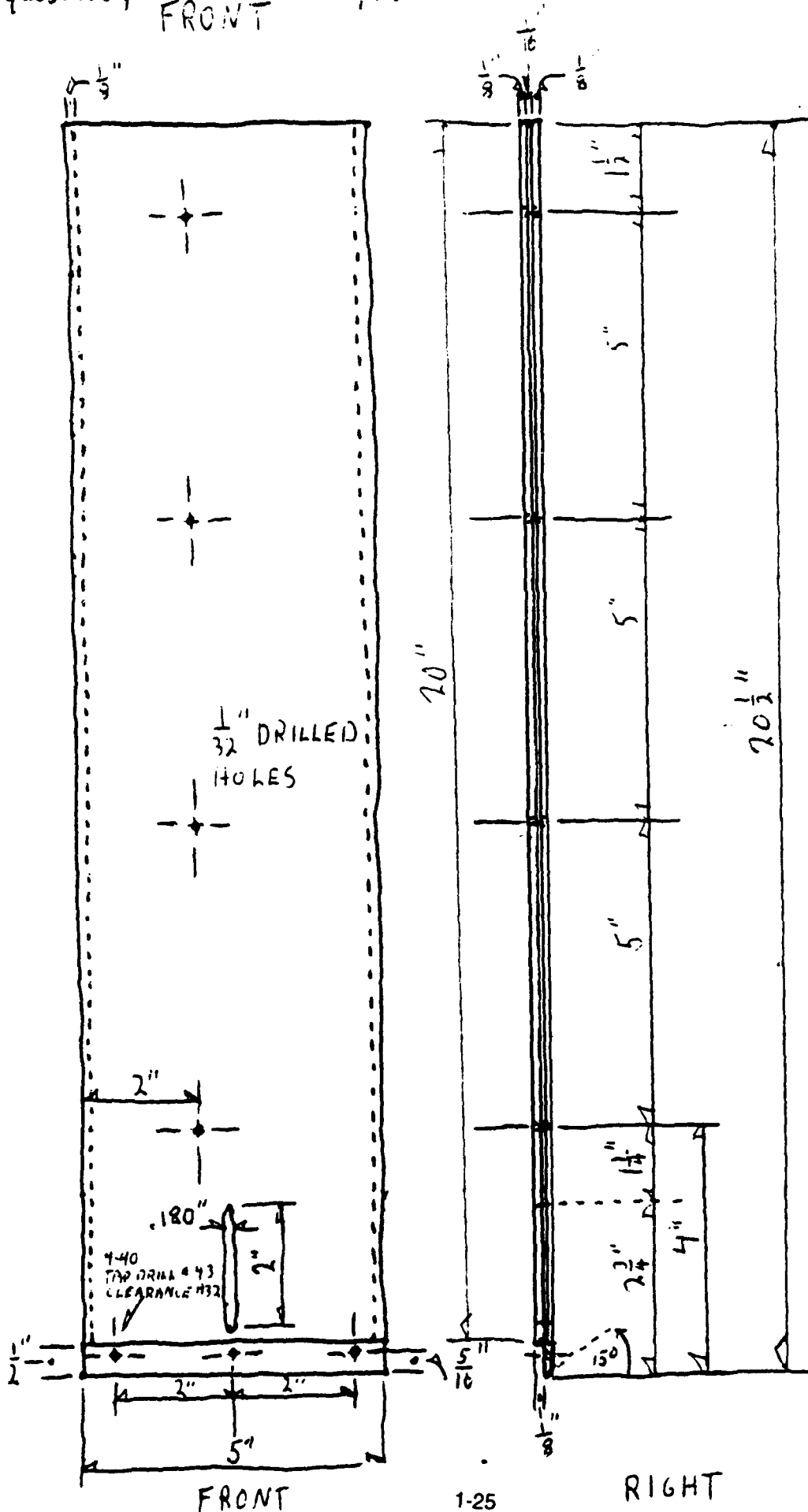


RIGHT

SCALE: $\frac{2}{5}" = 1"$

SKETCH: RIGHT PIVOTAL WALL OF WIND CHIMNEY

-any questions; call Lee Bunch, 255-6250
FRONT

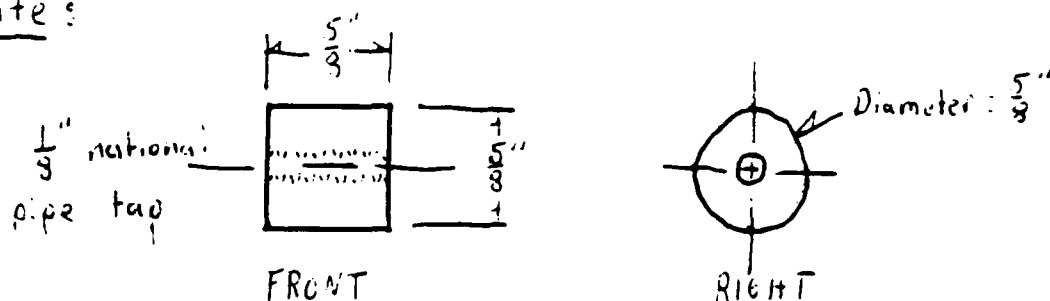


SCALE: $\frac{2}{3}" = 1"$

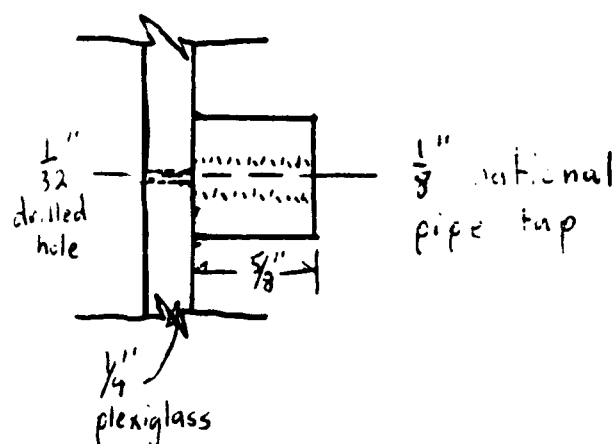
SKETCH: Separate / Attached $\frac{5}{8}$ " Bosses of Right & Left Walls

- any questions; call Lee Bunch 255-6250

Separate :



Attached :



SCALE: FULL SIZE

NOTE: - 8 Bosses should be constructed from 5" of a $\frac{5}{8}$ " diameter piece of plexiglass rod

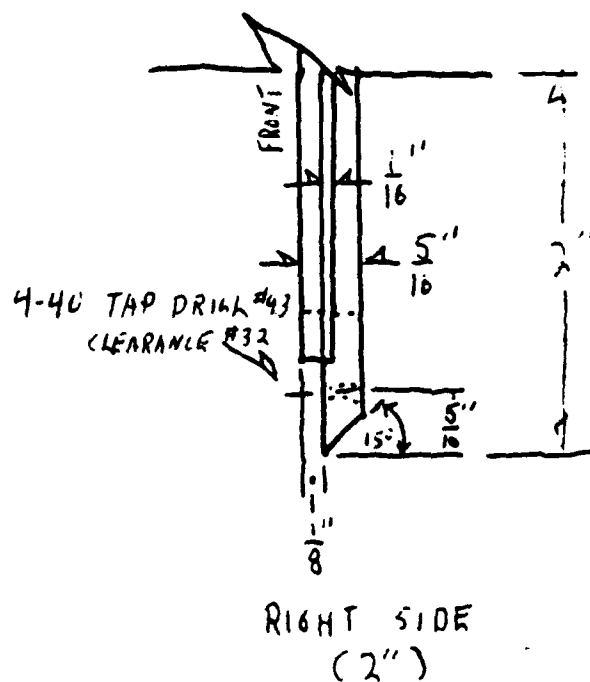
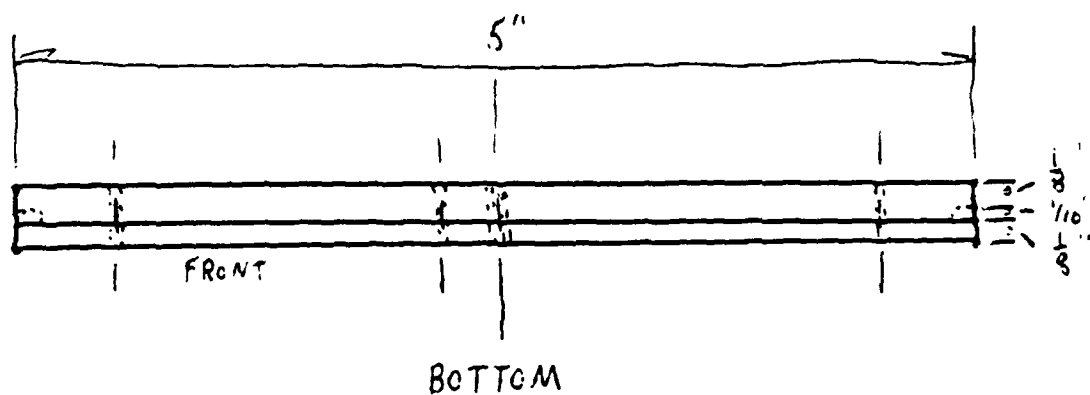
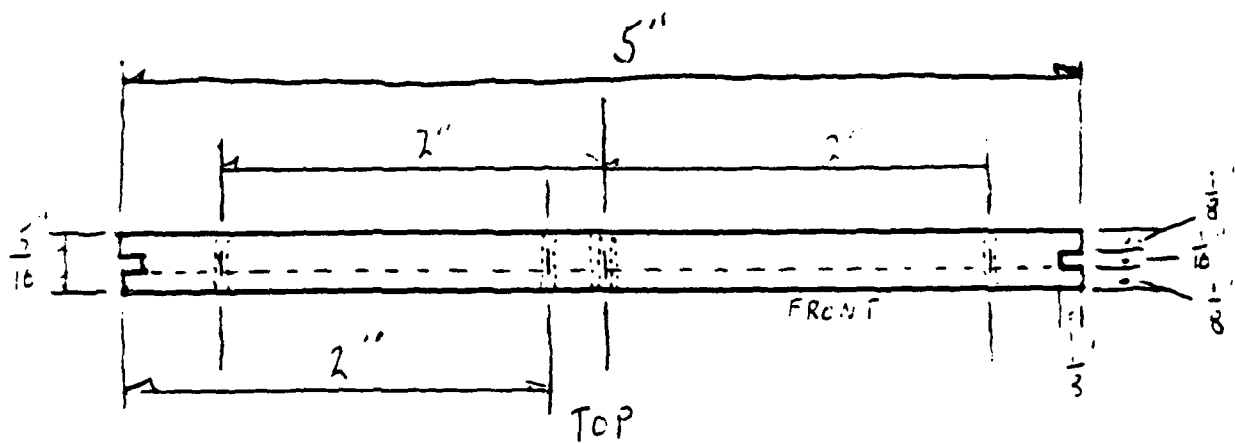
- Drill $\frac{1}{8}$ " national pipe taps into bosses

- Attach bosses centered over the 8 $\frac{1}{32}$ " drilled holes located off center on the the left & right walls

* Secure bosses using methylene chloride or dichloromethane or something similar to insure no leaks and maintain a tight seal. Caulk around seals if necessary for extra seal.

SKETCH: TOP/BOTTOM/RIGHT SIDE (2") VIEWS OF RIGHT PIVOTAL WALL

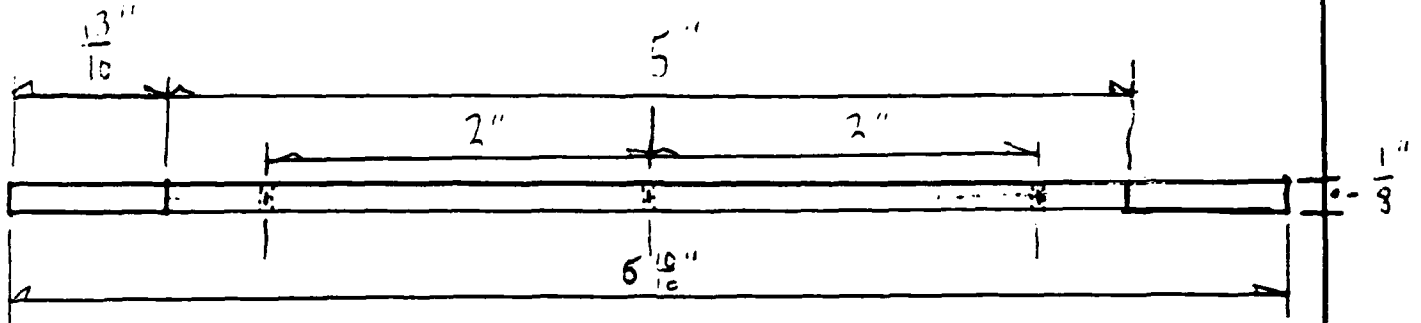
-any questions; call Lee Bunch, 255-6250



SCALE: FULL SIZE

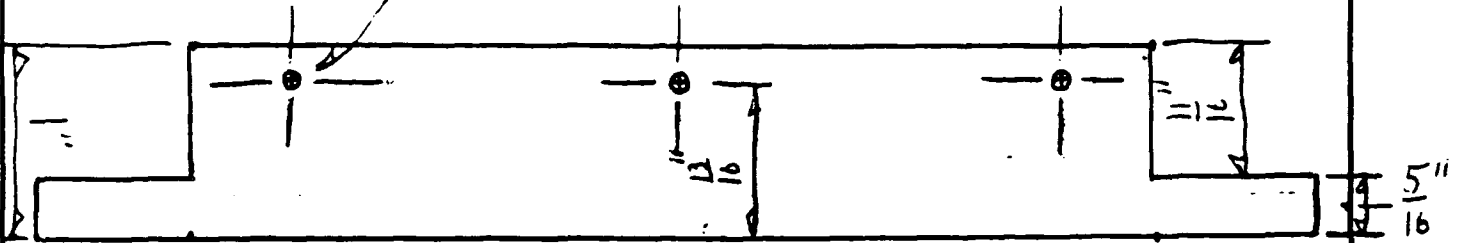
SKETCH: SEPARATE RUBBER HINGE FROM RIGHT PIVOTAL WALL

-any questions; call Lee Burch; 255-6250

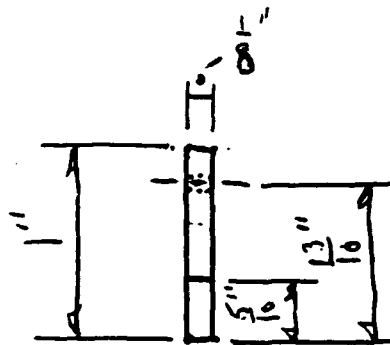


TCP

4-40 TAP DRILL #43
CLEARANCE #32



FRONT

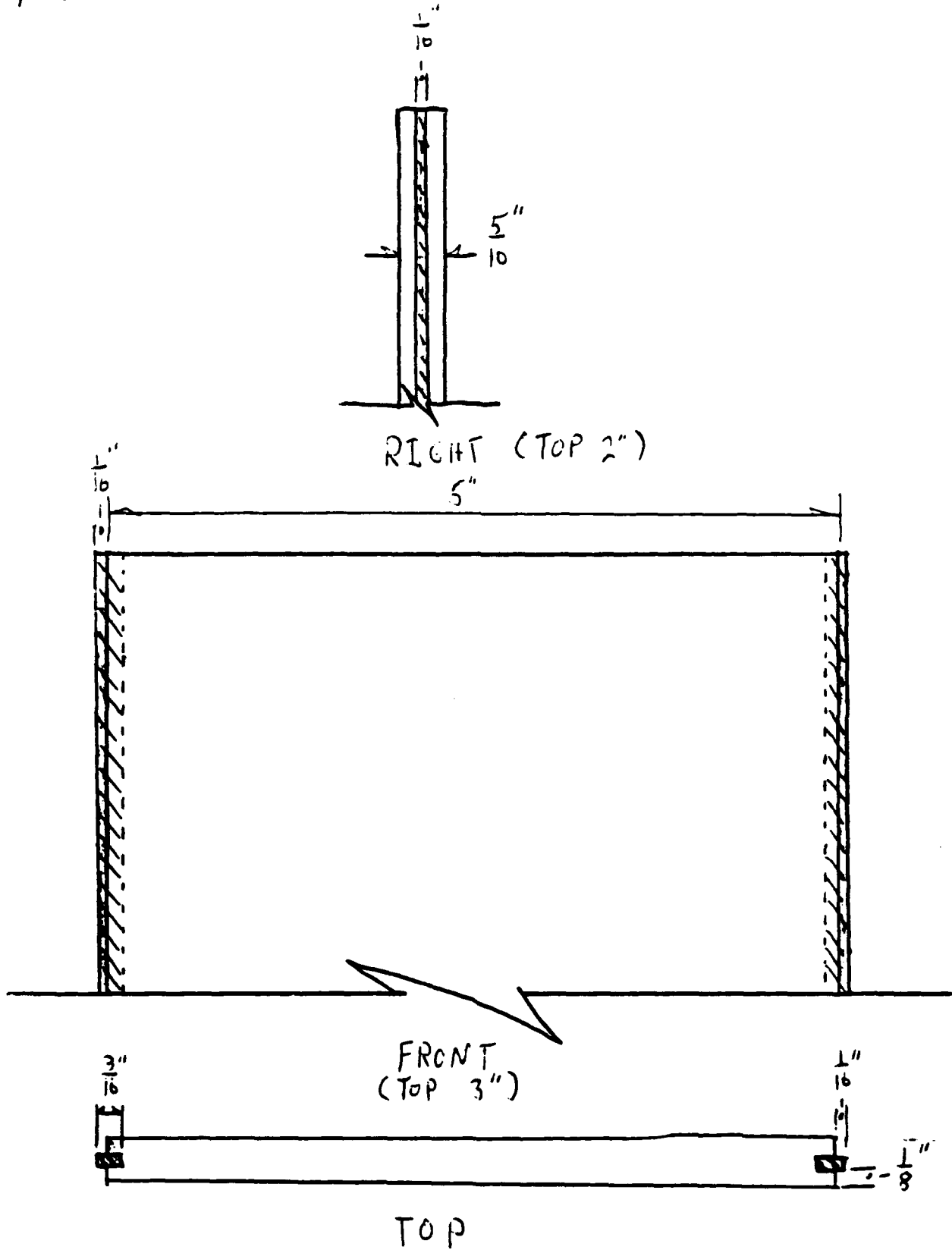


RIGHT

SCALE: FULL SIZE

SKETCH: RUBBER SEALS ON RIGHT PIVOTAL WALL

-any questions; call Lee Bunch, 255-6250

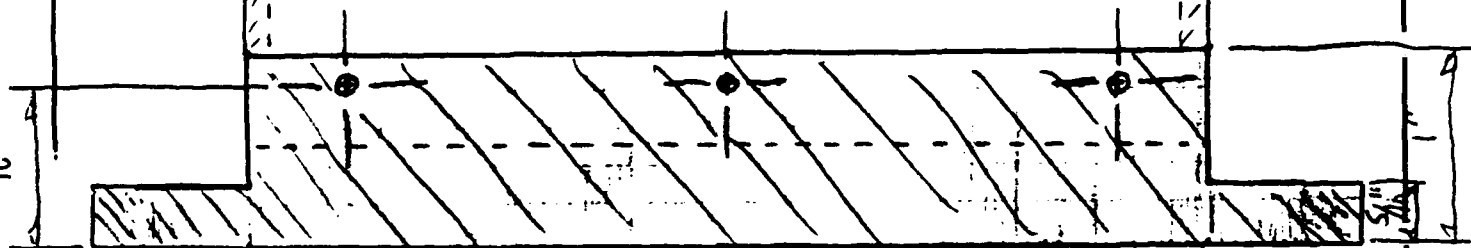
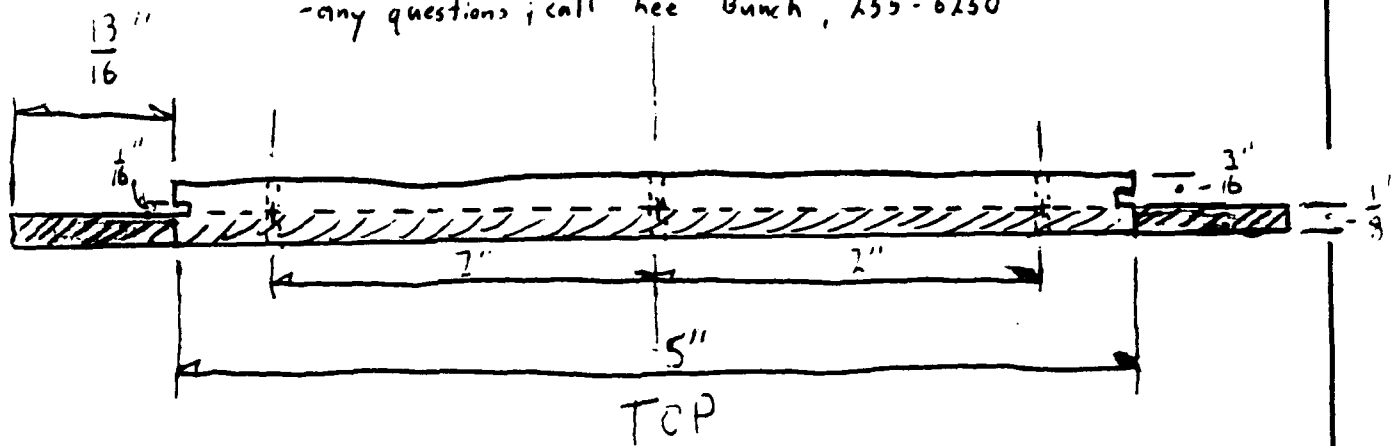


SCALE: FULL SCALE

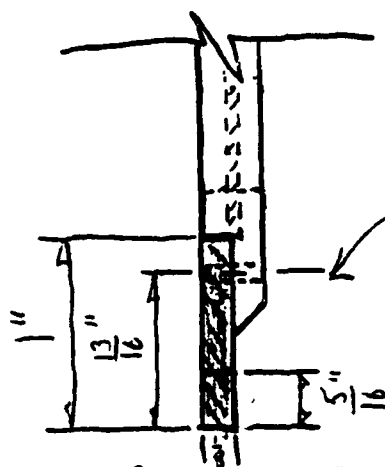
■ = RUBBER (exposed)
 ▨ = RUBBER (hidden)

SKETCH: RUBBER HINGE ON RIGHT PIVOTAL WALL

-any questions; call Lee Bunch, 255-6250



FRONT (BOTTOM 2")



RIGHT (BOTTOM 2")

4-40 TAP DRILL #43
CLEARANCE #32

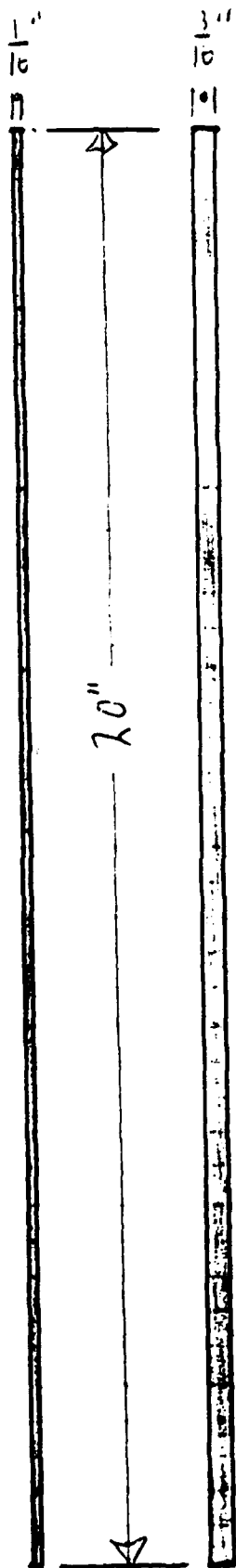
SCALE: FULL SCALE

1-30

■ = RUBBER (exposed)
▨ = RUBBER (hidden)

SKETCH: SEPARATE RUBBER SEALS FROM RIGHT PIVOTAL WALL

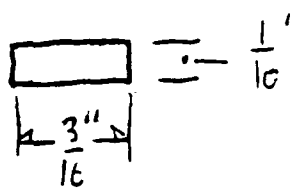
- any questions; call Lee Bunch, 255-6250



RIGHT

FRONT

MAGNIFIED TOP VIEW



SCALE: $\frac{5}{16}'' = 1''$

1 BOX = $\frac{1}{16}''$

SCALE: $\frac{1}{5}'' = 1''$

**THE PRODUCING AND TESTING
OF SINGLE CELL
THERMAL BATTERIES**

**HIGH SCHOOL SUMMER APPRENTICE
TINA M. CHILCOAT**

ABSTRACT

The testing of single cell thermal batteries provides information about larger scale batteries. Numerous Differential Scanning Calorimeter tests were performed to ensure that the materials used were well characterized. The testing of the vanadium glass showed that although the results were encouraging, more research and testing must be done in the future in order to find a lightweight, long lasting, high voltage thermal battery.

INTRODUCTION

An important part of today's expanding technology is the use of thermal batteries in missiles and other weapons. These batteries can function over a range of lifetimes and voltages depending upon the components of the battery. At Wright-Patterson Air Force Base Battery Lab, single cell batteries are prepared and tested to determine the possible voltage and lifetime of a larger scale battery made of like materials. Decisions upon the chemical makeup of batteries are made based upon the results of these tests.

DISCUSSION OF THE PROBLEM

Finding a lightweight material that will produce greater

voltage over a longer period of time is an important consideration for a thermal battery. Greater voltage in lightweight materials will lessen the overall weight of the weapon that uses the battery. One of the objectives of the Battery Lab is to discover the right combination of materials to improve battery performance.

RESULTS

Several laboratories and various equipment are used to produce single cell batteries in the battery lab. First, anode, cathode, and electrolyte, also commonly called anolyte, materials must be produced in the dry room, which is a room in which the atmosphere is kept as water-free or dry as possible. The dry room is used due to the reactive nature of the components of the cell with water. Certain percentages of anolyte to anode and anolyte to cathode are mixed separately in twin-shell blenders. For most single-cell batteries, in the battery lab, lithium-aluminum is used as the anode, and a lithium chloride and potassium chloride mixture is used as anolyte. The catholyte material may vary from battery to battery.

Due to the ever present problem of power outages at the base and mechanical difficulties in the dry room, a necessity arose to test the material for the presence of water. A Differential Scanning Calorimeter, or DSC, was used to perform the test upon the anolyte material. Using a small sample of

the material, the DSC checks it for the presence of unwanted, foreign components, by measuring the heat capacity of the material versus a known standard. When several changes in the heat capacity occur, it is due to the boiling or melting of some components of the material. When only one change in the heat capacity occurs and it is at the proper melting point of the material, it demonstrates that the material has been formulated properly. However, if the melting point is too far away from the expected result, the material may not be what it is believed to be.

As shown in several DSC runs on pages 7, 8 and 10, a lithium chloride/potassium chloride anolyte melts at approximately 352°C. As shown on page 7, the melting point of the material is determined by connecting the flat before and after plateaus with a straight line. This line is crossed by another line which is drawn continuous with the trace of the downward dip of the run. At the point where those two lines cross, a line is dropped perpendicularly down to the temperature scale to determine the melting point of the material. A similar DSC of a similar anolyte is shown on pages 8 and 10.

On page 9, an example of a lithium chloride/potassium chloride anolyte containing water is shown. Using the same procedure as described above on the first dip, the boiling point is determined to be around 100°C which matches water most of the time. However, on some occasions, the water

molecules may be very strongly attached to the material and might not boil at the proper temperature. Due to the presence of water, the material was considered contaminated. Further contamination is caused by the reaction of water with part of the material. Its melting point and the melting point of the actual material have also been shown on the DSC run.

A normal DSC run of lithium chloride/potassium chloride anolyte is shown on page 10. Some of this anolyte was allowed to sit out in the "wet" air for a relatively short period of time. The running of a DSC, shown on page 11, shows the results of allowing this to happen. Also, the water contamination appears to be worse than that on page 9 due to the longer sharper drops. The difference between these two DSC runs properly show the necessity of keeping all cell materials and cells in dry air.

A DSC may also be run to see that a material is what it is believed to be. For example, on pages 12 and 13, the run of a sample of vanadium glass, used as a catholyte in several runs, shows that the material is indeed a glass. Glasses characteristically have one plateau followed by a second plateau. These two runs insure that the cell testing results will be true for this particular vanadium glass.

After DSC testing and the blending of the anode and catholyte materials, pellets must be made for the anolyte, anode and cathode. A press is used to make three gram pellets of the cathode and anolyte materials and two gram pellets of

the anode material. These pellets are then stacked with the anolyte in between the anode and cathode. The three pellets are then placed between two dumbbells which have metal strips to provide contacts for the reading of voltage. The anode side is marked to distinguish it from the cathode side.

The cells are carried in a desiccator to the single cell tester and passed through a vacuum chamber into the tester. The computer and heaters are set so that testing may begin. The cell is placed between two sheets of mica and put in position between the heated platens. The ram comes down, and hopefully, the connections are good. The cell test could run for anytime from a few seconds to over half an hour based upon the temperature.

Most of the test runs have been of vanadium glass. On page 14 is a plot of three different glasses run at 400°C. In spite of the not so wonderful start of vanadium glass cathode number twenty-nine shown in red, the three curves seem to show close resemblance in their runs. At around six minutes, a characteristic voltage drop to about 2.2 volts is exhibited.

On page 15 are some of the vanadium glass test results performed at 425°C. All appear to follow each other closely, especially the curves printed in green and violet which are both of vanadium glass cathode number twenty-five. This close resemblance shows that these two runs were especially accurate.

On pages 16 and 17 are plots showing the tests of

vanadium glass cathodes number twenty-seven and twenty-nine at various temperatures. The tests of cathode number twenty-seven on page 16 show similarity in the traces for 400°C, 425°C, and 450°C. However, the trace for 525°C died quickly showing that this temperature was too hot for the cell and burned it out. The tests of vanadium glass cathode number twenty-nine on page 17 do not appear as similar as those for vanadium glass cathode number twenty-seven.

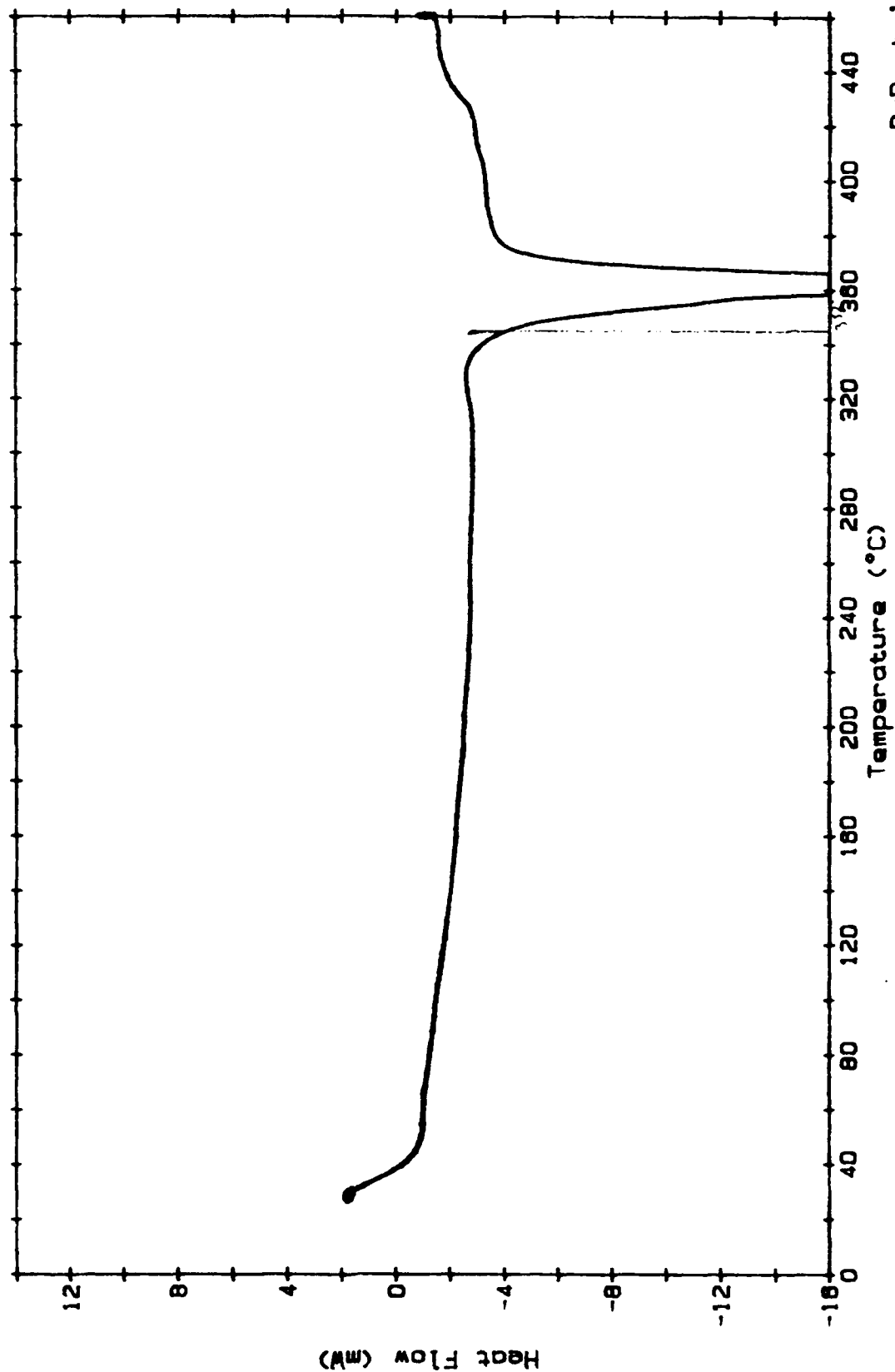
Also tested were niobium and molybdenum compounds. These compounds did not produce as much voltage or last as long as the vanadium compounds. Examples of the molybdenum tests are shown on page 18.

CONCLUSIONS

The search for a longer lasting, higher voltage, lightweight thermal battery will continue. Although, the results of the vanadium glasses at 400°C and 425°C are impressive and encouraging, other batteries, on the market already, are even better. The research of the battery lab will have to continue forever in their search due to the ever constant need to improve on the technology of the past and push on towards the future.

Sample: ANOLYTE 21
Size: 5 MG
Rate: 20 DG/MIN

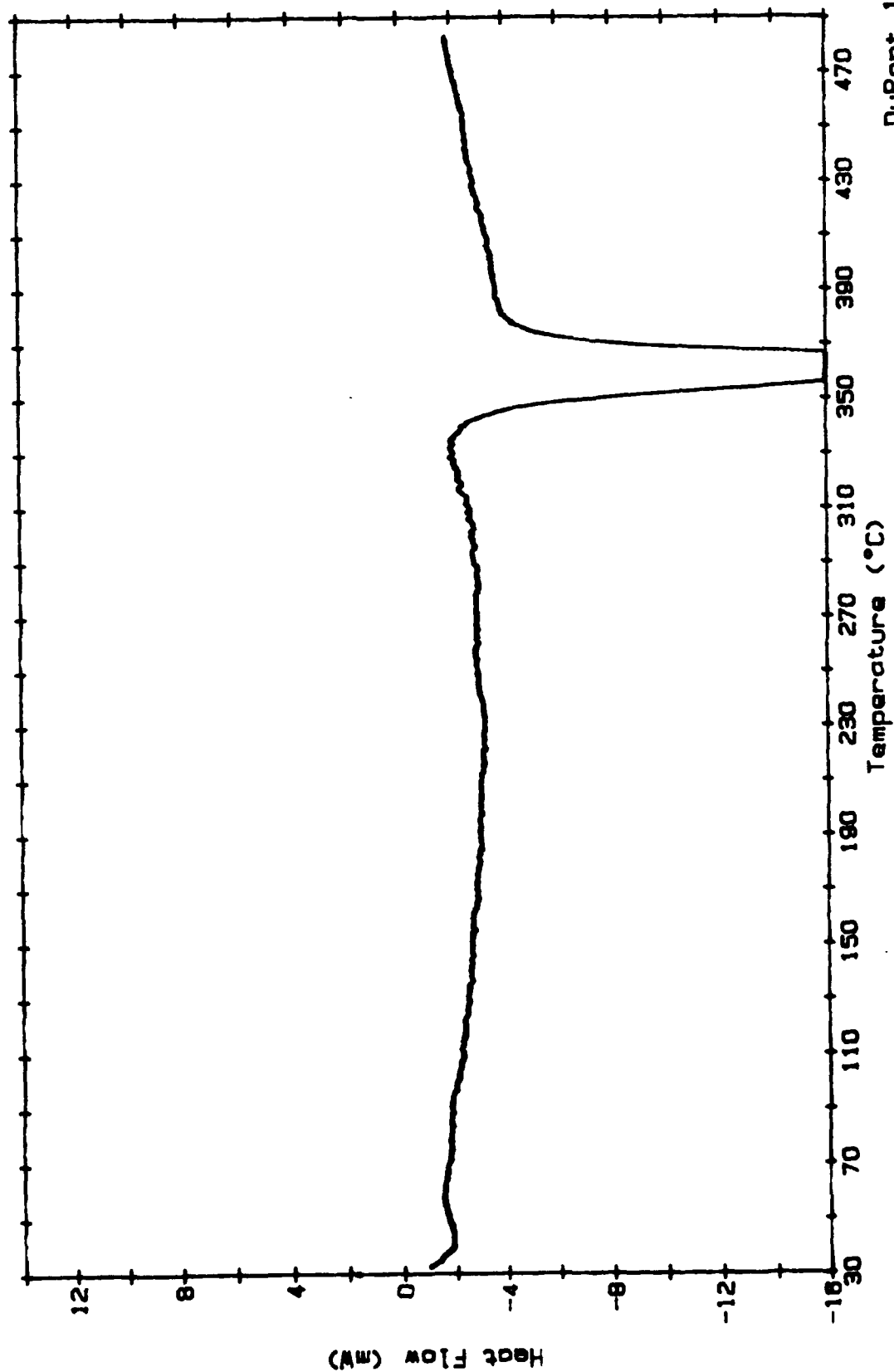
DSC



DuPont 1090

Sample: ANOLYE 23
Size: 6 MG
Rate: 20 DG/MIN

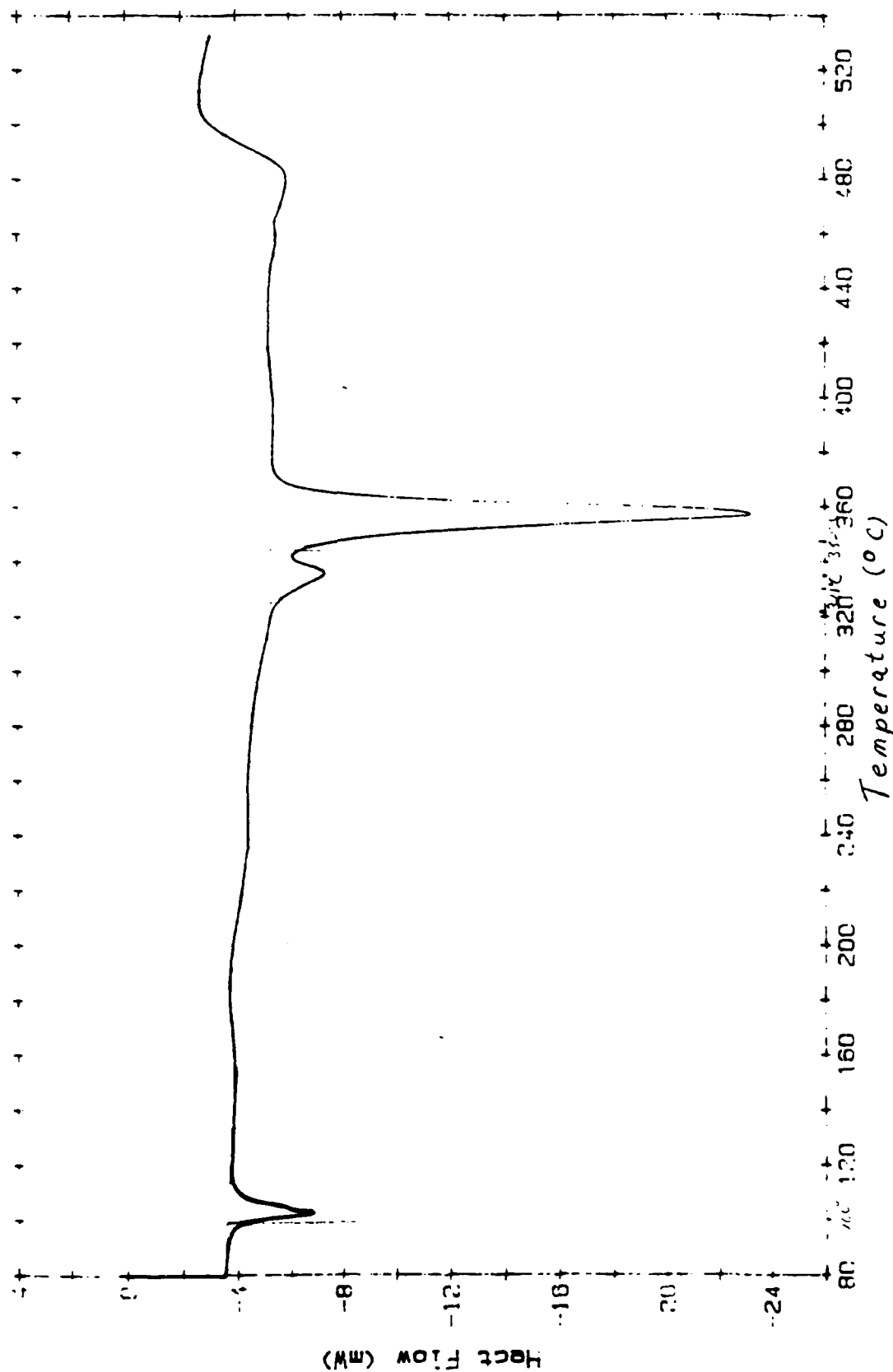
DSC



DuPont 1090

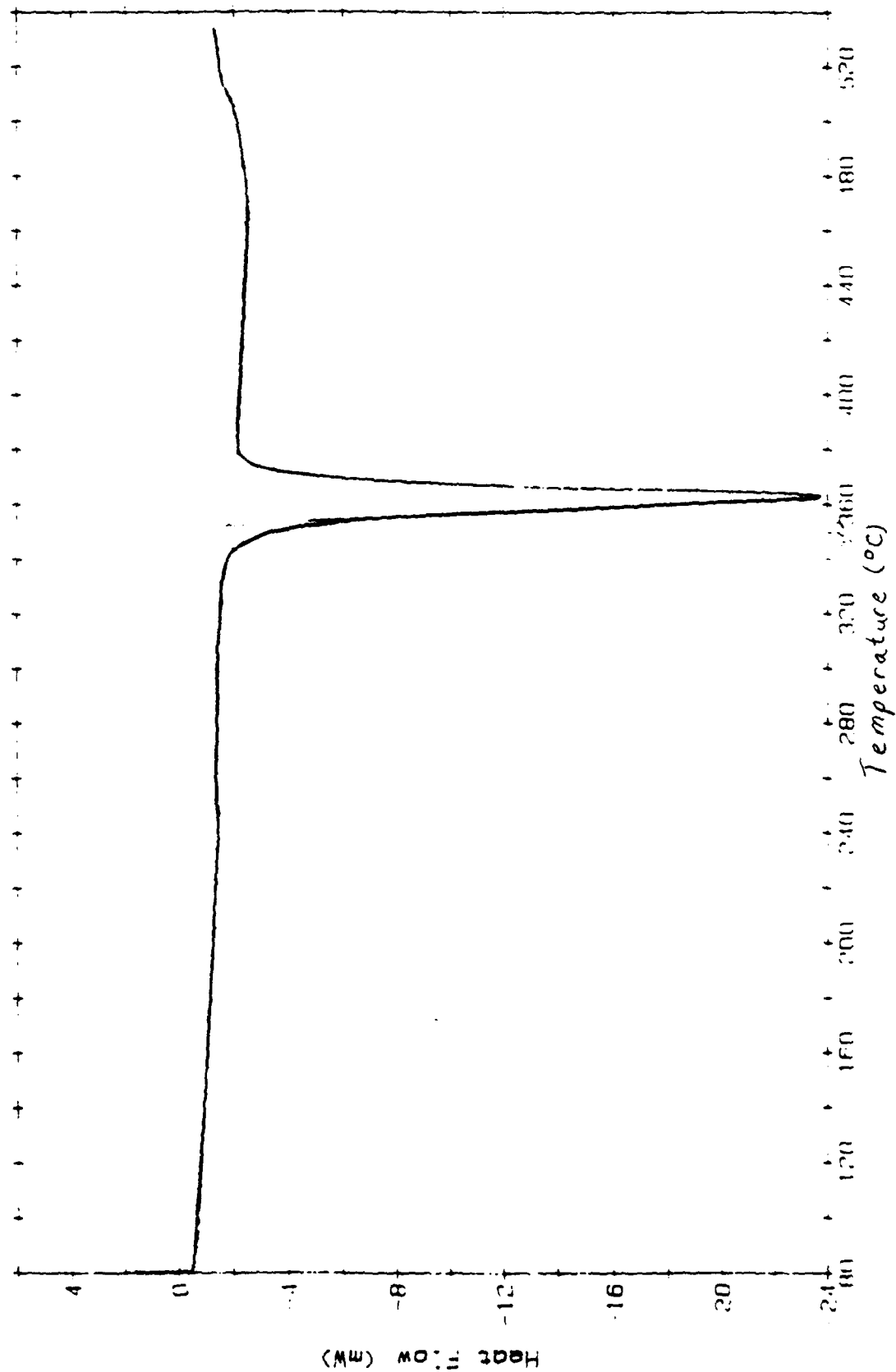
Sample: 2A 170
Size: 7 MG
Rate: 20 DEG/MIN

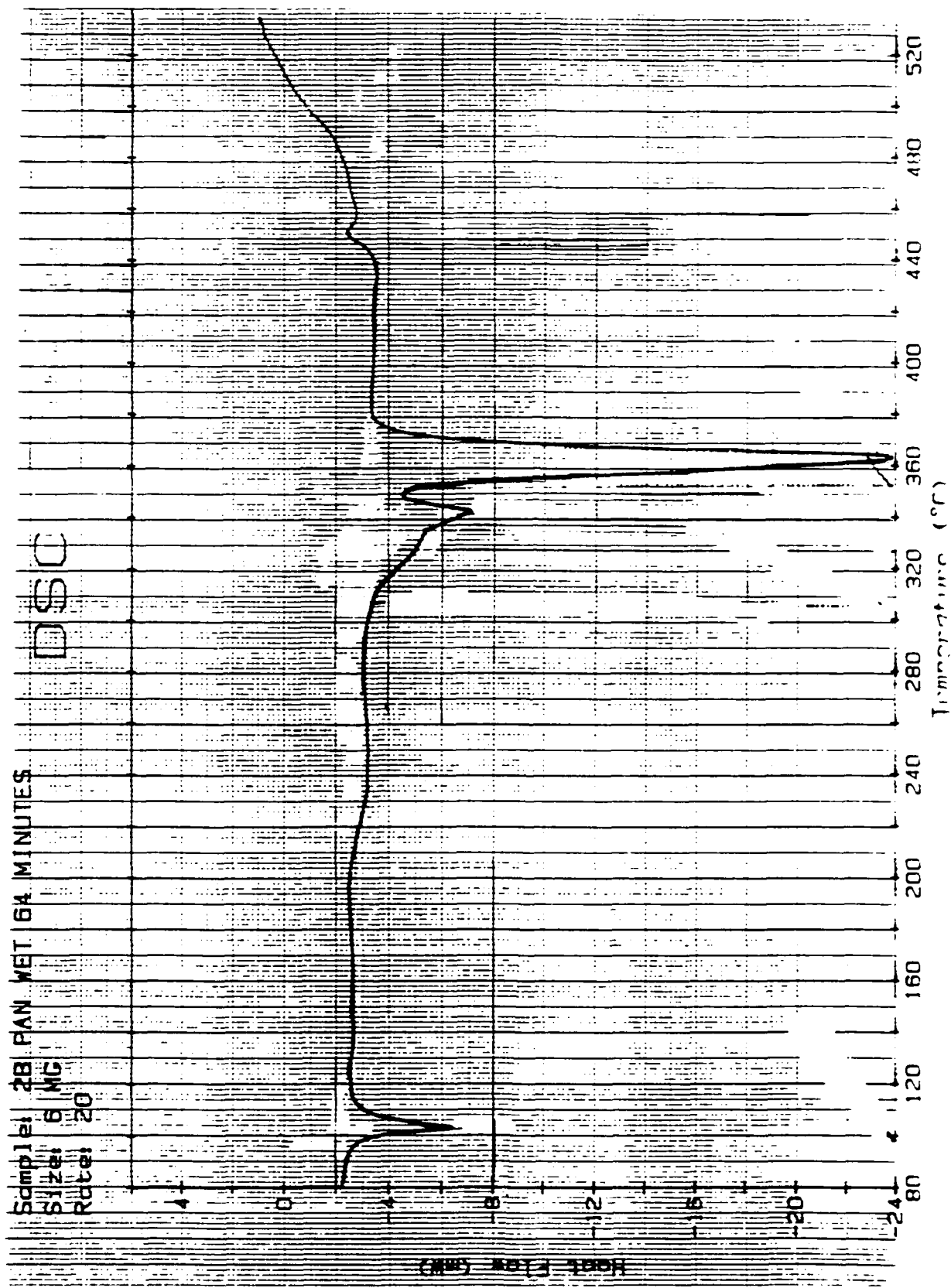
1060



Sample: 2B PAN DRY
Size: 5.5 MG
Rate: 20

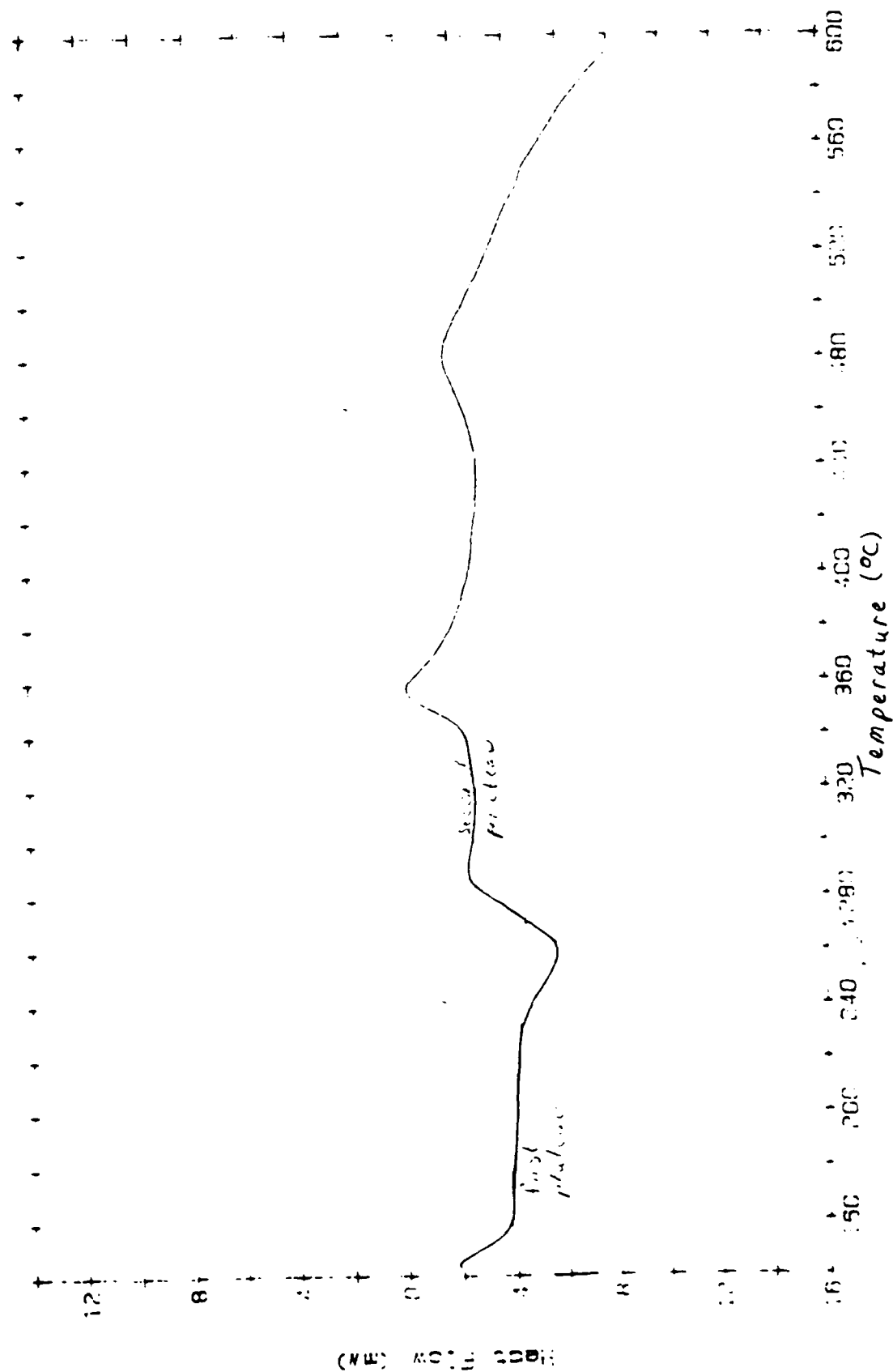
DSC

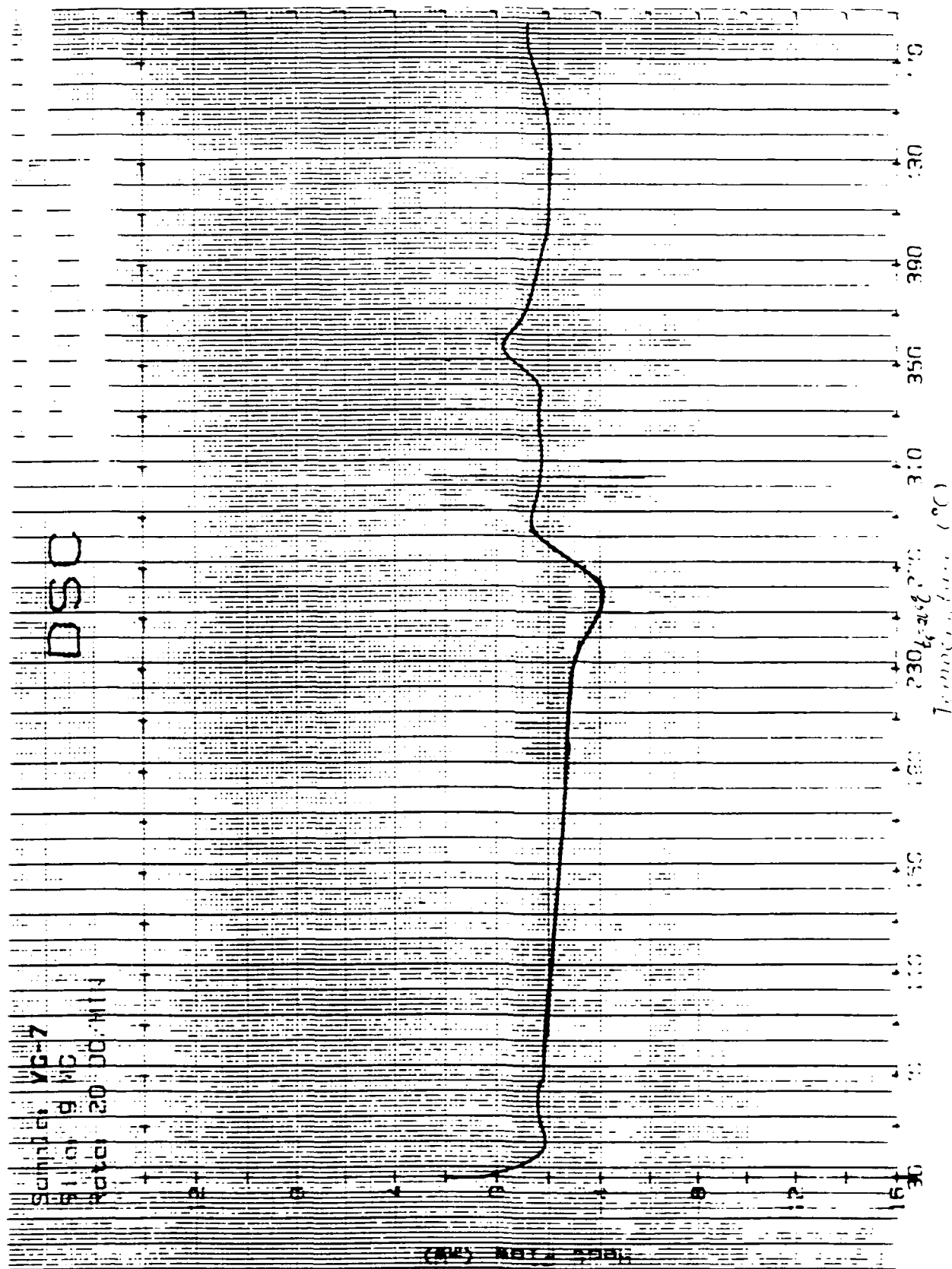




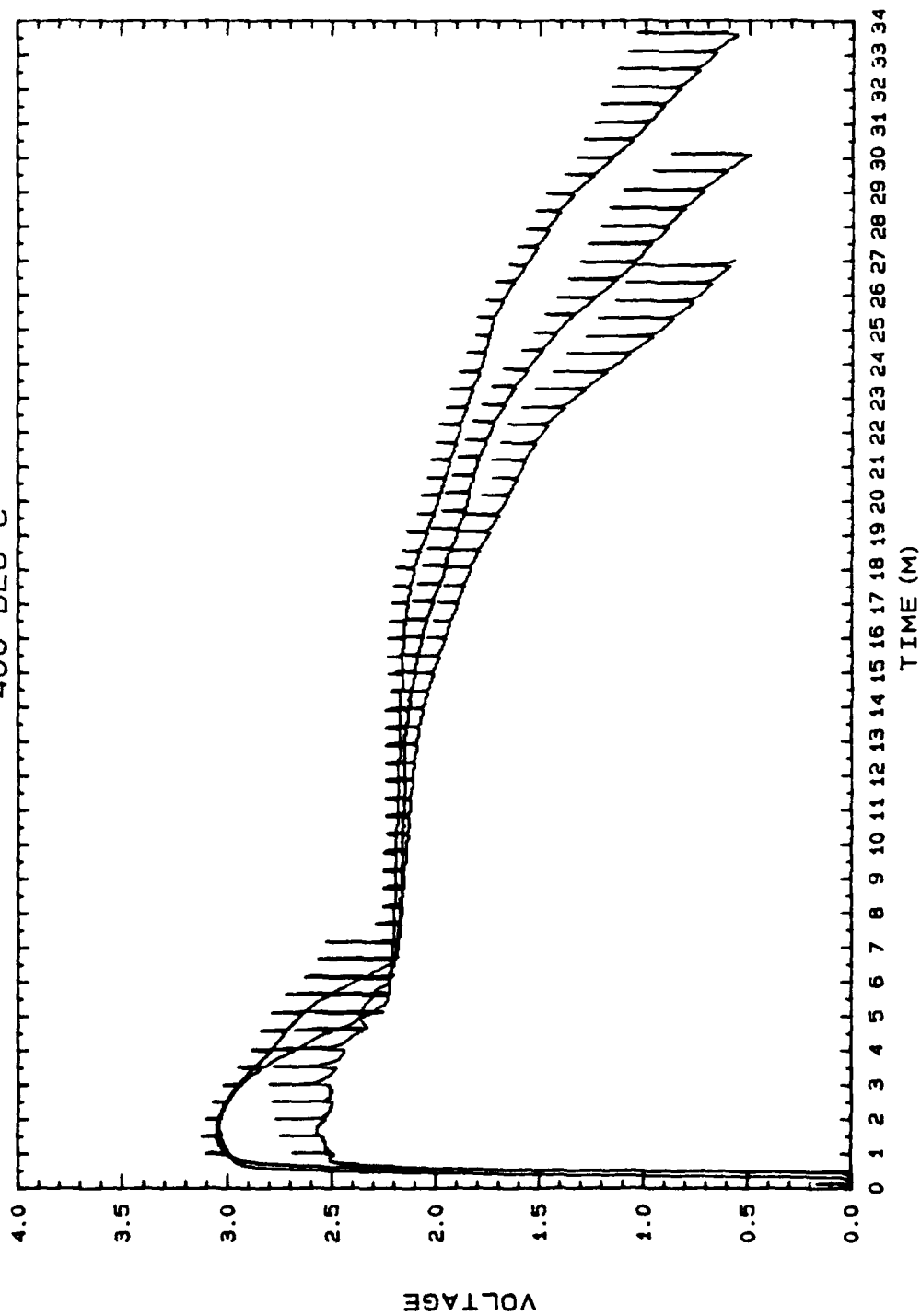
Sample: VG-7
 Size: 11 MG
 Rate: 20 DG/MIN

DSC





VANADIUM GLASS
400 DEG C

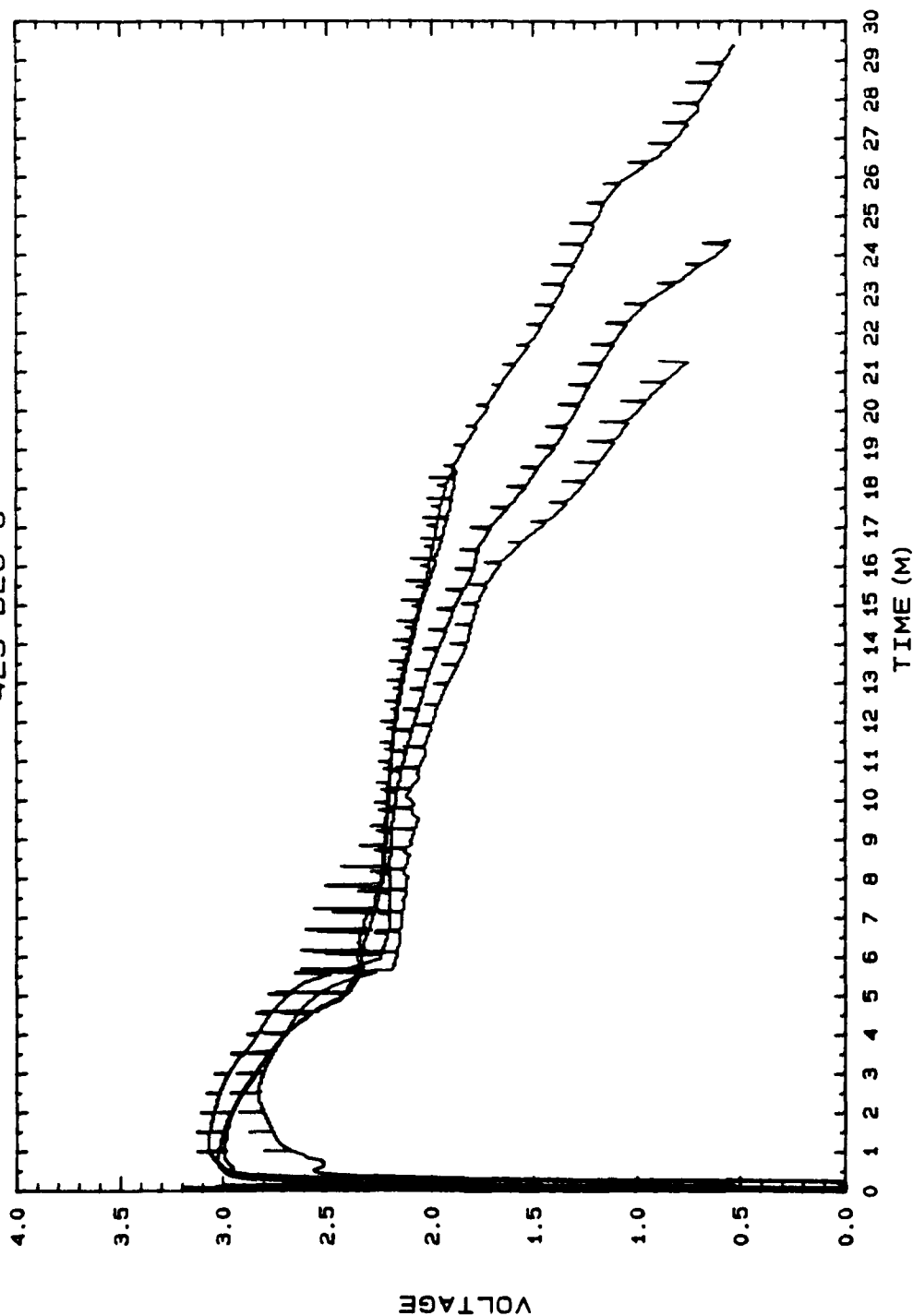


T07052: : D3
CRNT: 1.00 AMP
910705400002
L1CLVGLASS
AN#5
ANOL#23
CA#28

T07051: : D3
CRNT: 1.00 AMP
910705400001
L1CLVGLASS
AN#5
ANOL#23
CA#29

T07055: : D3
CRNT: 1.00 AMP
910705400005
L1CLVGLASS
AN#5
ANOL#23
CA#27

VANADIUM GLASS
425 DEG C

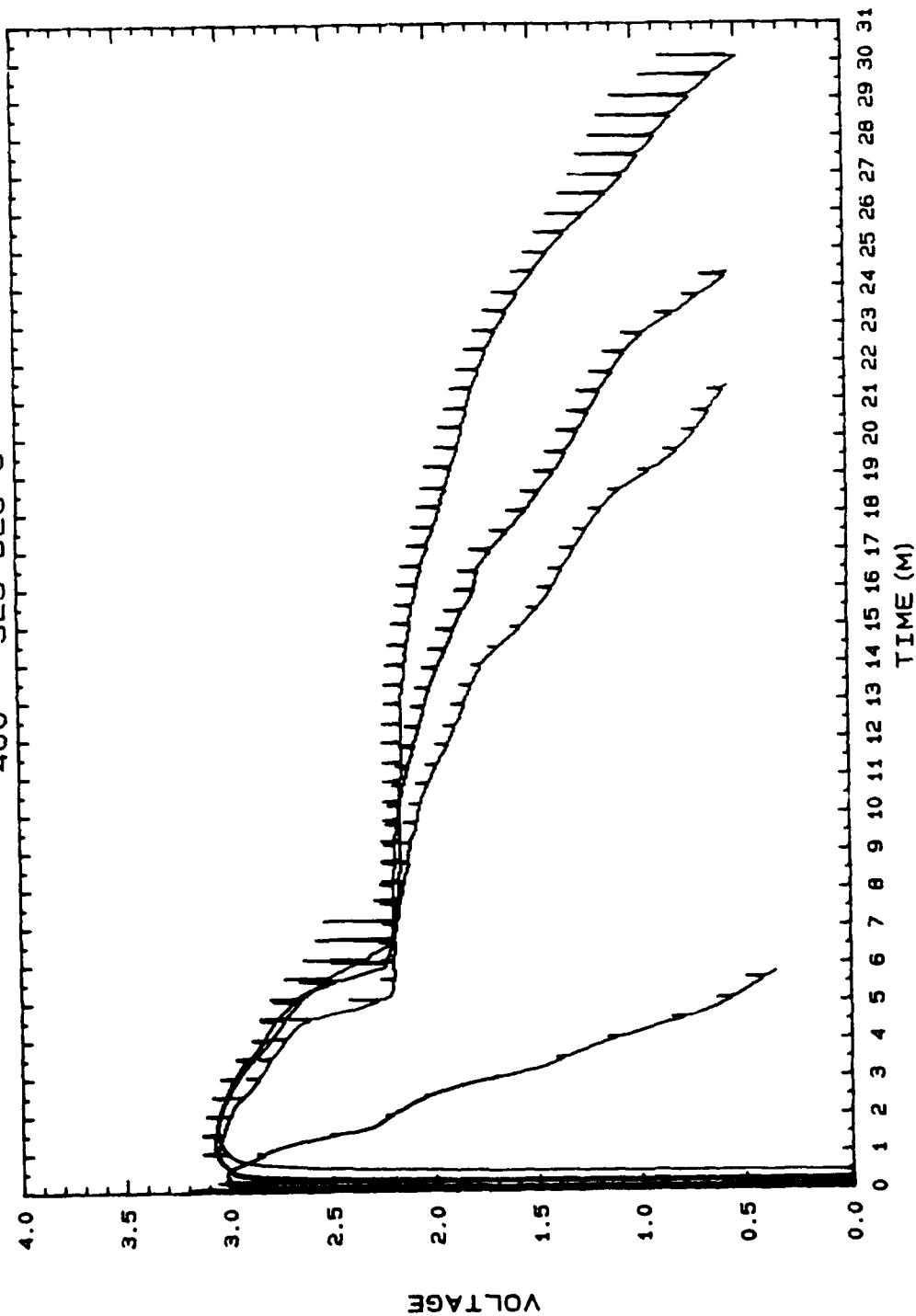


T07023:: D3
CRNT: 1.00 AMP
910702425003
LICLVGLASS
AN#5
ANOL #23
CA#27

T07024:: D3
CRNT: 1.00 AMP
910702425004
LICLVGLASS
AN#5
ANOL #23
CA#25

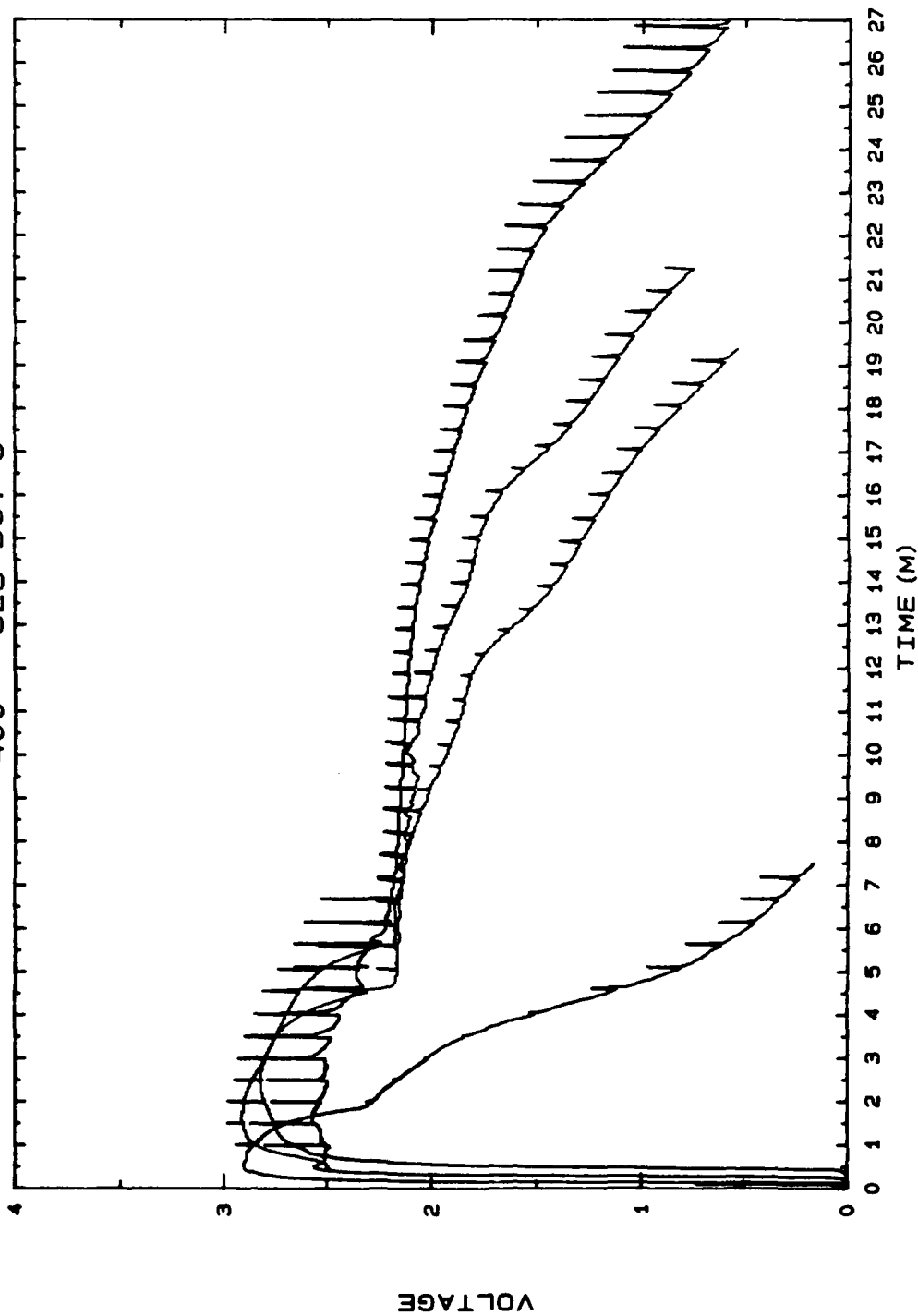
T07025:: D3
CRNT: 1.00 AMP
910705425006
LICLVGLASS
AN#5
ANOL #23
CA#25

VANADIUM GLASS
400 - 525 DEG C

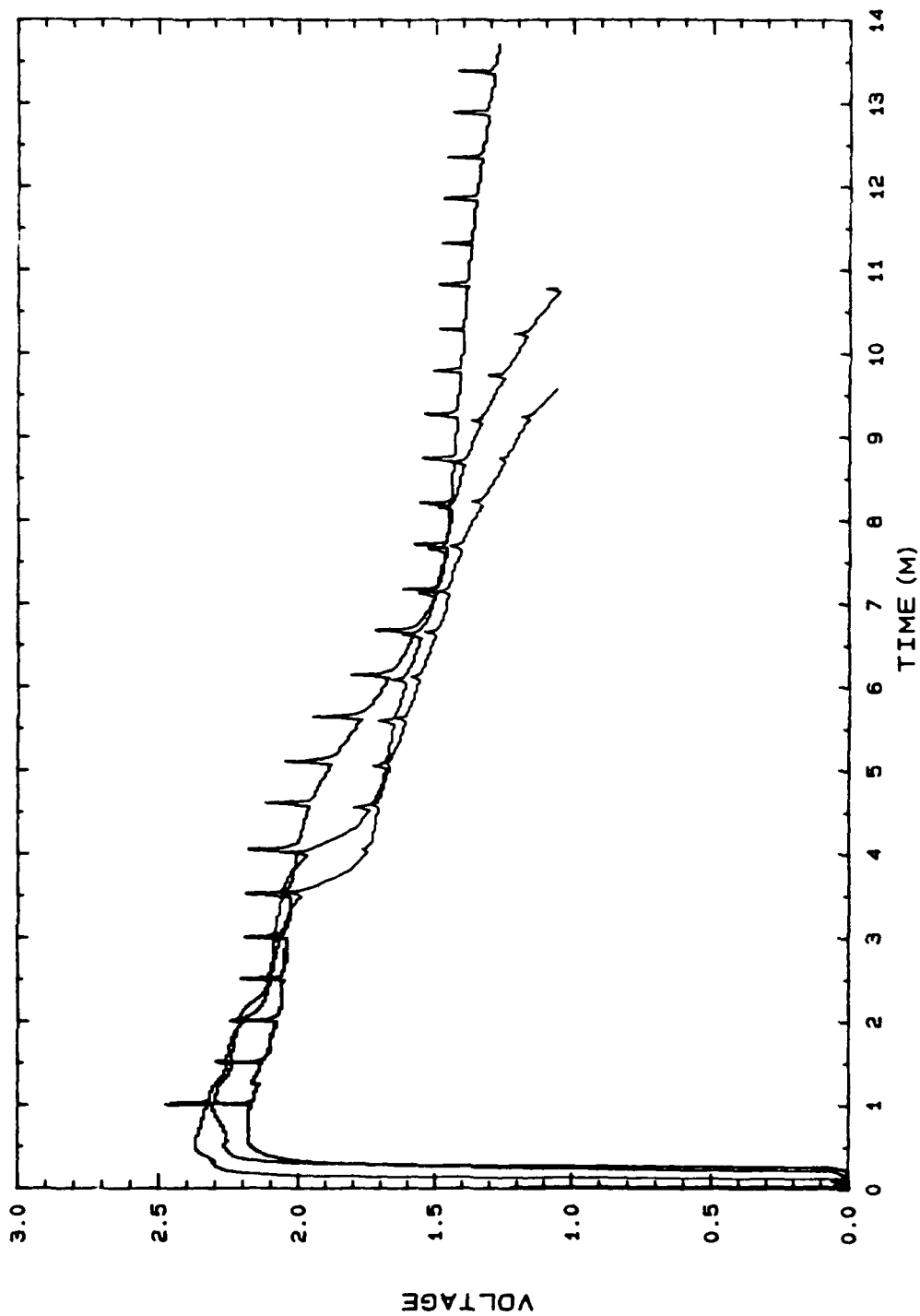


T07023: : D3 CRNT: 1.00 AMP 910702425003 LICLVGLASS AN#5 ANOL #23 CA#27	T07053: : D3 CRNT: 1.00 AMP 910705450003 LICLVGLASS AN#5 ANOL #23 CA#27	T07055: : D3 CRNT: 1.00 AMP 910705400005 LICLVGLASS AN#5 ANOL #23 CA#27	T07013: : D4 CRNT: 1.00 AMP 910701525003 LICLVGLASS AN#5 ANOL #23 CA#27
---	---	---	---

VANADIUM GLASS
400 - 525 DG. C



T07051::D3	T07031::D3	T07021::D3	T07011::D4
CRNT: 1.00 AMP	CRNT: 1.00 AMP	CRNT: 1.00 AMP	CRNT: 1.00 AMP
910708400001	910703450001	910702425001	910701525001
LICLVGLASS	LICLVGLASS	LICLVGLASS	LICLVGLASS
AN#5	AN#5	AN#5	AN#5
ANOL#23	ANOL#23	ANOL#23	ANOL#23
CA#29	CA#29	CA#29	CA#29



T07124: D3
CRNT: 1.00 AMP
910712400004
L1CL-MO
AN#6
ANOL#23
CA#32

T07126: D3
CRNT: 1.00 AMP
910712475006
L1CL-MO
AN#6
ANOL#23
CA#32

T07125: D3
CRNT: 1.00 AMP
910712450005
L1CL-MO
AN#6
ANOL#23
CA#32

A STUDY IN SOLID AND LIQUID LUBRICANTS

Jeffrey S. Cropp

For my summer apprenticeship, I worked at the Aero Propulsion and Power Directorate at Wright-Patterson Air Force Base, in the Lubrication Branch of the Fuels and Lubrication Division. The work I completed occurred in two sections. The first section lasted for one week and involved a study of the change in an oil's acidity and viscosity relative to a certain period of use in an engine. The second section, which consumed the following seven weeks, dealt with changes occurring in solid lubricants heated to various temperatures in both air and nitrogen atmospheres. The research into the solid lubricants was classified as Military Critical Technology, and therefore cannot be discussed in as much depth as I wished.

Various methods were employed in studying the solid and liquid lubricants. To determine the viscosity of the oils, the time elapsed as they flowed from one point to another in a calibrated viscometer was measured. The tubes were placed in 40°C and 110°C baths to determine their respective viscosity. The acidity of the oil was determined by a titration of the oil with a small amount of potassium hydroxide, and expressed as the Total Acid Number (TAN).

The study of the solid lubricants developed in three phases—thermal analysis, infrared spectra analysis, and evolved gas analysis. To begin with, a sample would be heated and a Differential Thermal Analysis (DTA) would be recorded. An example of the DTA of a standard compound is shown in Figure 1.

The DTA reveals endothermic and exothermic peaks which provide the researcher with points at which the sample may undergo a change, and which should therefore be investigated to a greater extent. The sample is heated once again, and a Thermogravimetric Analysis (TGA) is recorded. Figure 2 illustrates the TGA of a standard compound. The TGA displays the change in weight percentage which occurs over a known temperature gradient, and show the temperatures at which a major sample change occurs. During a TGA run, the evolved gas analysis is recorded, in the form of either a Linear Histogram or through Multiple Ion Monitoring (MIM). A linear histogram is a scan-by-scan record of the intensities of gases using molecular mass numbers. MIM shows the progression of gases evolved during a TGA run. The residue present after a thermal analysis is compressed into a thin, round wafer (called an "IR pellet") and scanned in both the mid- and far-IR using a Nicolet 740 spectrometer. The spectra provides the researcher with clues as to the composition of the changed sample. This is determined by the size of the peaks at various wavelengths. Figure 3 shows the absorbance peak of tungsten trioxide in the mid-IR, and Figure 4 shows the absorbance in the far-IR. All of these methods are important means for a researcher to develop a fairly good understanding of chemical changes occurring at different stages in the sample's heating.

The problems I researched were essentially the same for both types of lubricants. They basically involved the investigation of the extent to which each type of lubricant would change as a

result of wear or temperature. The liquid lubricants were examined to determine the effects of a certain number of hours of use in an engine on them. The solid lubricants were studied to determine their compositional changes induced by temperatures. An important question raised in addition to that of the degree to which both types of lubricants changed was exactly what compound was doing the lubricating at various temperatures if the original solid lubricant was indeed changing.

The answer to the degree to which the lubricants were changed was different for both types of lubricants. With the liquid lubricant, the acidity increased and the viscosity decreased with increased numbers of hours of use in an engine. The primary reason for this appeared to be the degradation of the lubricant due to oxidation. The solid lubricants, by contrast, generally changed composition greatly as a result of high temperatures in an oxidizing atmosphere (air). The degree to which this occurred depended on the specific solid lubricants and/or any catalysts present.

As the results indicated, the solid and liquid lubricants reacted differently in response to the tests which they were subjected to. Despite an increase in acidity and a decrease in viscosity, the liquid lubricant did not achieve a drastic change in its composition, except for the addition of oxidative byproducts or the depletion of an additive. The solid lubricants experienced many changes in their composition. The degree to which this occurred depended on the temperature and the

oxidative/inert atmosphere. Higher temperatures and air caused oxidation and created the largest and most frequent changes in the solid lubricants. Lower temperatures failed to induce significant changes in the solid lubricants, and no oxidation could occur in nitrogen. Due to the extent of change, it can be deduced that the original sample was not responsible for lubrication at high temperatures, which is therefore caused by whatever the sample becomes changed into. The only to properly determine what the final product might be is through the use of the IR spectra, which provided many clues to the identities of many of the samples.

Figure 1.

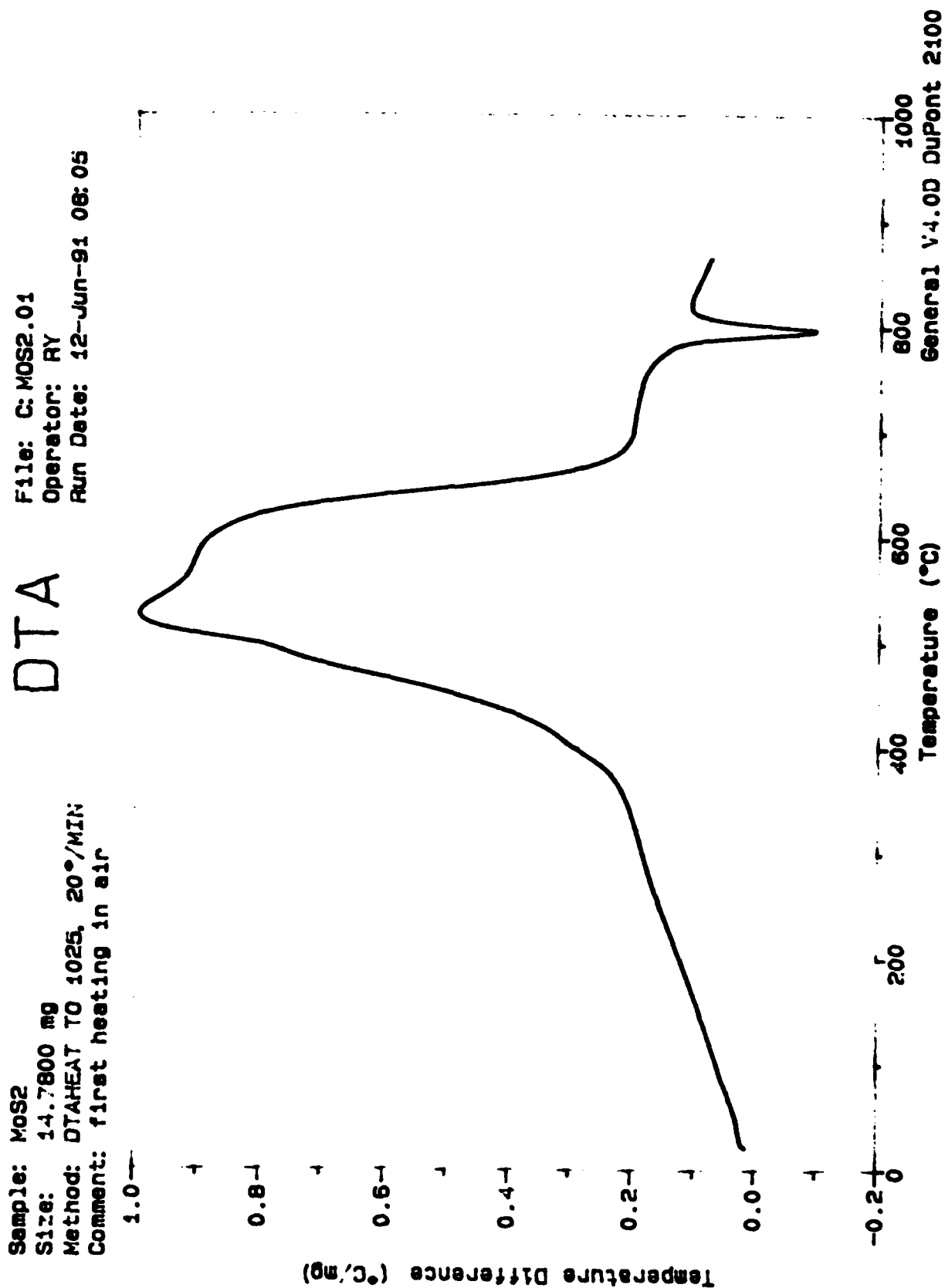


Figure 2.

Sample: MoO2
Size: 69.2130 mg
Method: 20, MIN TO 1050, ISO
Comment: AIR

TGA

File: C:\DRFM\002.30
Operator: DRF
Run Date: 16-Jul-90 09:29

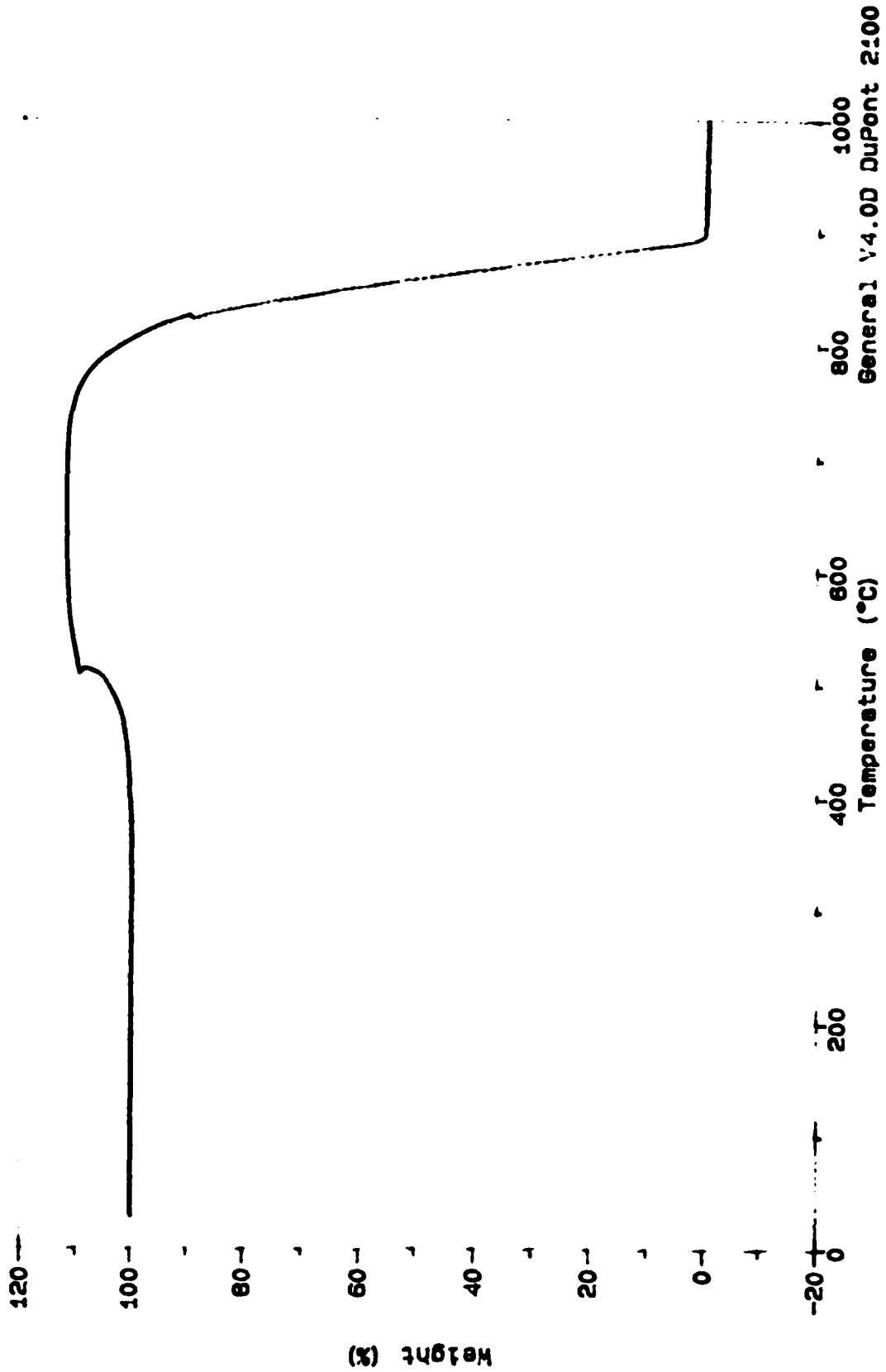


Figure 3.

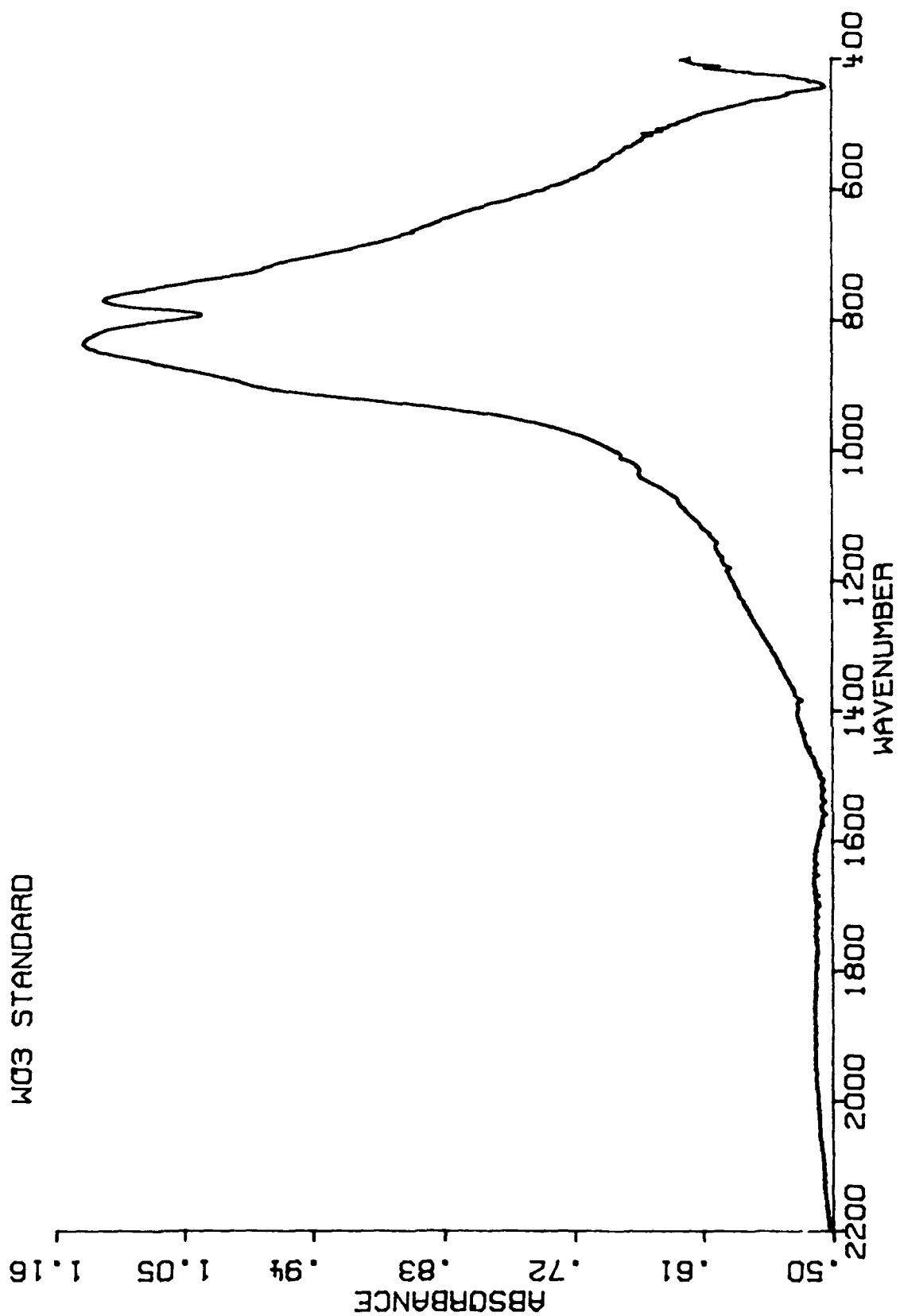
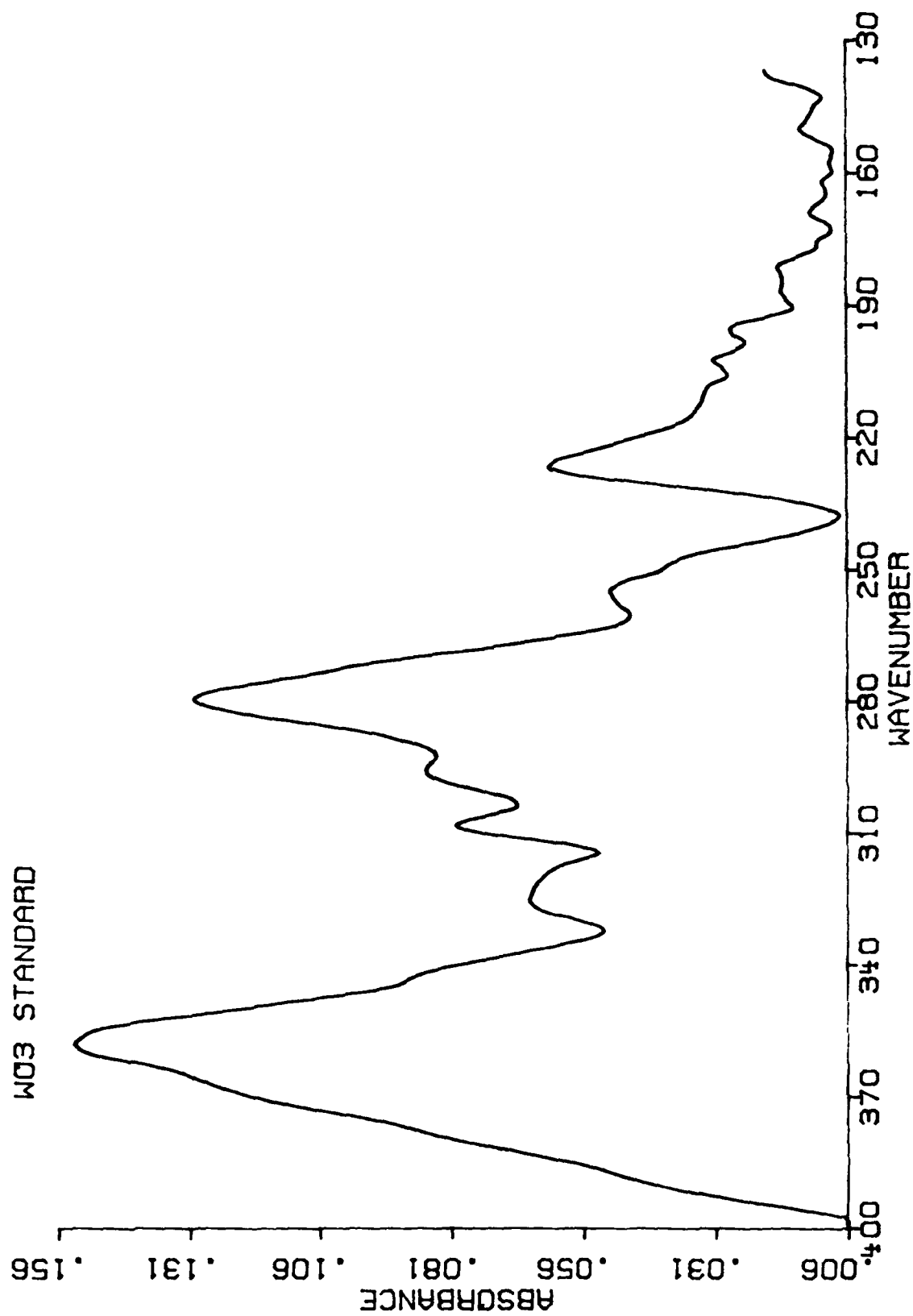
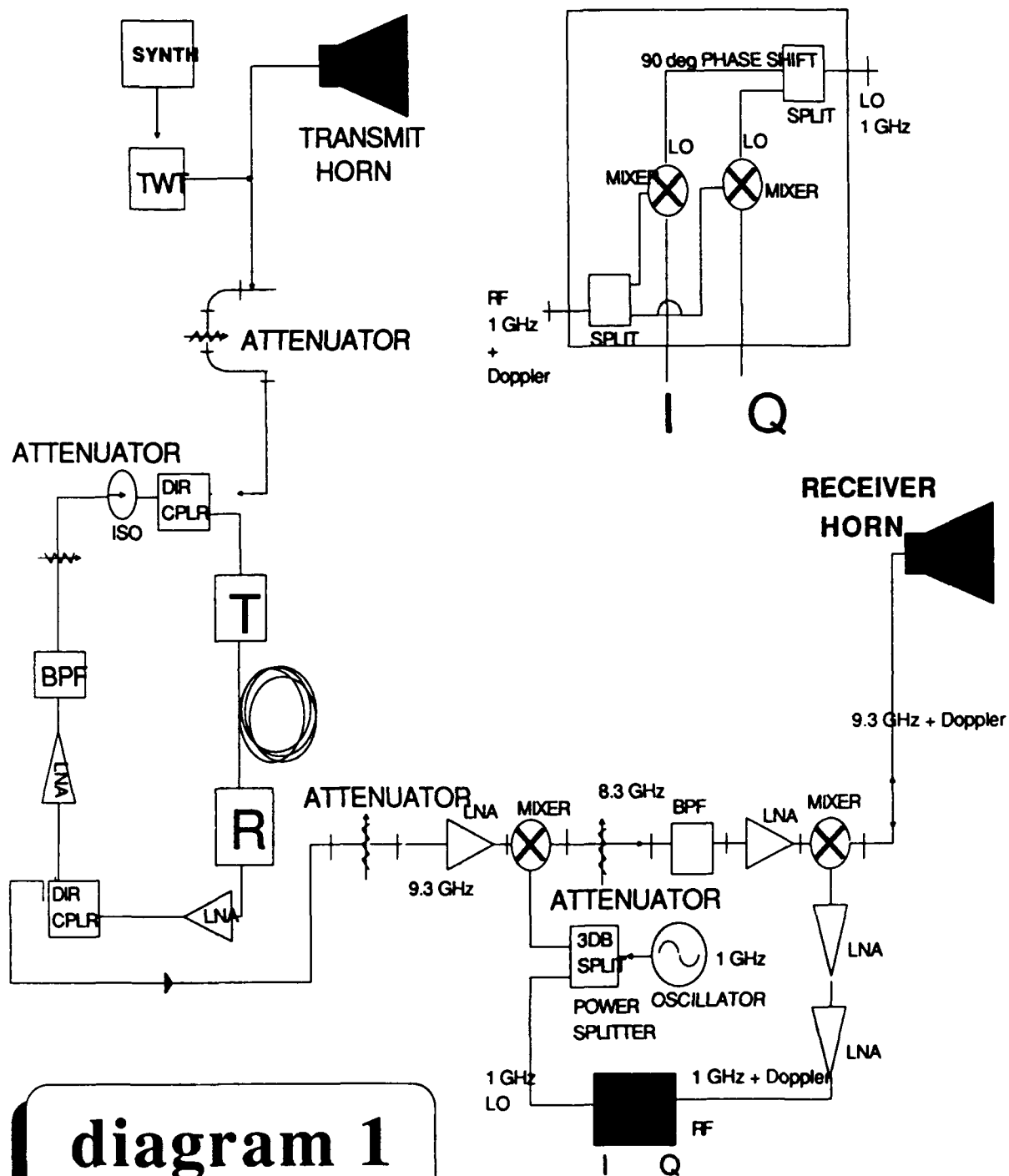


Figure 4.





FUEL RESEARCH

FUEL RESEARCH Matt Grider

The time period that this paper covers is from June 17 to August 9, 1991. During this period I have worked in the fuels branch of the fuels and lubrication division of Wright Laboratories at Wright Patterson AFB. I have been exposed to experimental techniques and equipment as well as the atmosphere of a research and development oriented workplace. The research I did was concentrated in the area of the thermal stability of jet fuel.

The Air Force wants to provide the best possible fuel for use in its aircraft and this task is the responsibility of the fuels branch. One of the major problems that can cause the degradation of fuel is oxidation. One effect of oxidation on fuel is that it decreases its thermal stability. The Air Force wants to use a fuel that is very thermally stable but will not be too expensive so it created the JP-8+100 additive program. This program is an ongoing research initiative that hopes to be able to increase the thermal stability of JP-8 jet fuel by putting additives into the fuel.

The degradation of jet fuel is a very complex issue because jet fuel is composed of many compounds and many pathways exist for the degradation to occur. When stored for

long periods of time fuels may react with oxygen to form peroxides. Peroxides attack rubber seals and other materials and can also adversely affect the thermal stability of the jet fuel. In order to prevent oxidation from occurring, an antioxidant is added to the fuel. The antioxidant is usually a hindered phenol or a butyl phenol derivative. Antioxidants only work to prevent oxidation so they must be added to the fuel before any oxidation is allowed to occur.

Copper and some other metals will reduce the thermal stability of a fuel and cause other problems associated with oxidation because these metals act as a catalyst for oxidation reactions. So a metal deactivator additive is often added to fuels and especially to those fuels that have undergone a copper sweetening process. The metal deactivators react with the metal to form a chelate and the chelate will not act as a catalyst for oxidation reactions.

The thermal stability of a jet fuel must be maintained because an unstable fuel will form gum deposits which can clog areas and cause engine parts to malfunction. To help maintain the thermal stability of a fuel, Jet Fuel Additive Number 5 (JFA-5) can be added to the fuel. JFA-5 helps maintain the oxidative stability and it also has a dispersive quality that helps prevent degradation products from sticking to one another to form a large enough particle to create problems. The composition of the different gums and solids that are formed is not completely known nor is the exact process of their

formation completely known. The reason for some of the tests is to try and determine what the deposits are composed of and if this could be determined then the reaction that takes place to form the degradation products may be determined and this will hopefully lead to the discovery of a process that will be able to lessen the formation of these products.

An experiment that was often run was an MCRT (micro carbon residue tester) stress test. In this experiment a sample of fuel is weighed and put into a vial and the vial is inserted into the MCRT which heats the fuel sample to 250 degrees celsius for various lengths of time. The stressed fuel and gum deposits are saved for further analysis and the vial is weighed again to determine the amount of solid deposit left by the fuel. The more thermally stable the fuel is the smaller the amount of deposit there will be. Using methylene chloride the gums from the MCRT can be dissolved and then run through a UV/vis spectrophotometer and the stressed fuels can be dissolved in hexane and run through the same apparatus. The UV/vis data showed that the degradation products may very well contain aromatic compounds because when the unstressed UV/vis spectra was subtracted from the stressed spectra the absorbance remained in the ultraviolet which is where aromatic compounds absorb. The gums also absorbed in the UV so aromatic content may be related to thermal stability.

The stressed fuel was also ran through an FTIR which measures absorbance in the infrared range. Using a computer

program the absorbance can be integrated with respect to a baseline for the area of the carbonyl peak where oxidation takes place and groups of carbon double bonded to oxygen are formed. The integrated absorbance areas show that the more thermally stable fuels have a smaller oxidation stretch area while those fuels less stable have a larger area.

Stressed fuel was analyzed using a refractometer and this shows that the more stable fuels have a lower refractive index than the less stable fuels after they have been stressed. An interesting phenomenon occurs in both the refractive index and infrared absorbance areas of the stressed fuels. The fuels stressed in the MCRT for two hours have values lower than both the one and three hour stressed fuels in these two tests.

The gums were run through a TGA(thermogravimetric analyzer) which measures percent weight loss against temperature. This test suggests that as the gums are further oxidized an oxidation product may form that initially will be heavy but as the temperature increases it will lose this weight. A comparison of the TGAs of the same gums shows that at 600 degrees celsius, TGAs run in nitrogen have 10 to 28 percent more weight remaining than TGAs run in air. Until 500 degrees celsius though the tests run in air have more weight remaining than the tests run in nitrogen but between 500 and 600 degrees celsius, TGAs run in air lose around twenty percent of their weight while those run in nitrogen lose only about two percent. The oxidation product of the gum therefore must

be less stable than the original gum because most of the weight has disappeared at the end of the TGAs run in air.

Another experiment run to stress the fuel is a flask test. The flask test consists of a silicon oil bath that is heated to 180 degrees celsius and two flasks containing thirty milliliters of the fuel sample and connected to cooled condensers. The flasks are lowered into the bath and oxygen is bubbled through the sample. After the flasks are removed from the bath and allowed to cool the fuel is filtered away from the gums which are then dissolved in acetone and filtered. The acetone is then driven off and the gums are collected. The filter paper will have collected the solids that are not soluble in the fuel or acetone. The results from this test show that different fuel formulations react to temperature stressing in different manners. In this test a 2827 formulation, which is less stable than 2747 in other tests such as the MCRT, formed fewer gum deposits than the 2747. The 2827, however, formed more solid insoluble deposits than the 2747. The 2827 also showed almost no carbonyl peak when it was run through the FTIR and these results illustrate the complexity of finding the most thermally stable fuel because the type of degradation products will differ between fuel formulations and depending upon the severity of stressing, one fuel may be more stable than another but then after more stressing appear less stable.

The Phoenix rig test is designed to closer simulate aircraft conditions to evaluate fuel stability. The Phoenix has a copper block in a cylinder with a piece of tubing as the test section running down the middle. The tubing is analyzed using the Leco carbon burnoff machine after a test to determine how much carbon deposit was formed. The tube is also weighed before and after the test. Different fuels at different flow rates can be used as well as replacing parts on the rig itself such as a filter to judge the quality of a piece of material or component that may be used. The amount of dissolved oxygen in the fuel can also be controlled. A gas chromatograph measures the dissolved oxygen content of the fuel at different phases in the rig and it will also measure methane gas content which would indicate that fuel pyrolysis has occurred.

The Phoenix rig also has been modeled. A computer program models the rig and allows different variables such as the tube test section size and fuel flow rate and others to be entered. The program will then go through its calculations and give the amount of deposits formed, oxygen depletion, and other figures. The actual numbers that modeling gives you may not be totally valid but even if the exact numbers are not perfect, modeling can be used to predict trends very well. The advantages of modeling are that it is cheaper, less labor intensive, and much quicker than the Phoenix rig.

Determining the most desirable jet fuel is a very complex issue. The thermal stability aspect of jet fuel is a

relatively new area in which much more research can be done and will continue to be done in the future as new test procedures are employed and new ideas are introduced that will eventually lead to the realization of the "perfect" jet fuel.

SUMMER HIGH SCHOOL APPRENTICESHIP PROGRAM

WRIGHT PATTERSON AIR FORCE BASE

AERO PROPULSION AND POWER LABORATORY

H-BAY

Mark Hansell

August 9, 1991

MENTOR: DR. JERRY BEAM

Project: Static Wicking

R. S. Gaugler came up with the idea for the heat pipe in 1944. His idea has grown and taken new shapes. In the beginning, a heat pipe's main purpose was to transfer energy to heat things like ovens. However, today heat pipes are mostly used to cool things like electronics.

Heat pipes are able to cool things efficiently because they take advantage of two natural properties: evaporation and condensation. A heat pipe consists of a closed system, usually a pipe as its name indicates, with a working fluid and its vapor at equilibrium. A porous wick structure, to which the liquid is attracted to or wets, separates the liquid from the vapor within the system. Heat enters one end, called the evaporator, and causes the liquid to evaporate. The vapor, then, leaves the liquid and joins other vapor in the adiabatic region. Here the vapor travels to the opposite end of the system called the condenser. At this point it loses its latent heat of vaporization and condenses onto the wick or other liquid. From here, the liquid is pumped back to the evaporator for another cycle. This pumping, called capillary pumping, is caused by the pressure difference between the liquid and the vapor across the wick structure.

Heat pipes are a very efficient way to transport heat, but they do have certain limitations due to their construction. The first limit is an axial limit called the sonic limit. It is a limiting factor because the vapor will flow to the evaporator faster when more heat is added. However, once the vapor reaches a certain velocity it cannot increase the amount of vapor or the

velocity at which it is flowing according to the mass flow equation. $m = \rho v A$.

where: m - mass flow rate

ρ - density of the liquid

v - velocity of vapor

A - area that vapor flows through

In steady-state operation the mass flow of the vapor will be equal to that of the liquid. The second limit is a radial limit called the boiling limit. This also limits the amount of heat entering the evaporator. If too much heat enters, nucleate boiling will begin, which for the most part is good for transferring energy. When the boiling interferes with the liquid returning to the evaporator, it will probably make the evaporator dry out. The third limit is another axial limit called the entrainment limit. This occurs when the vapor picks up droplets of fluid from the liquid/vapor interface and carries it to the condenser. This limits the amount of liquid returning to the evaporator and the amount of heat being transferred. The final limit is an axial limit called the capillary limit. This sets up a range of pressures, called capillary pressures, that will allow the heat pipe to run efficiently. When the capillary pressure is too low, it will not be able to overpower the pressure drops in the liquid, vapor, and as a result of gravity.

The highest pressure that will allow the pipe to run efficiently is called the maximum capillary pressure. The capillary limit is so important because it must satisfy this equation. $\Delta P_c \geq \Delta P_l + \Delta P_v + \Delta P_g$.

where: ΔP_c - capillary pressure

ΔP_l - liquid pressure drop from the
condenser to the evaporator

ΔP_v - vapor pressure drop from the
evaporator to the condenser

ΔP_g - pressure drop do to gravity

The capillary pressure is a function of both the surface tension of the liquid at the testing temperature and the wetting angle of the wick, in this case of the screen. The most important variable, however is the capillary radius, which is different for each screen. $\Delta P_c = 2\sigma \cos \theta / r_c$.

where: σ - surface tension of the liquid

θ - wetting angle of the screen(s)

r_c - capillary radius

The data found from the above equation should equal that of the below equation which can be used in practice.

$$\Delta P_c = \rho gh$$

where: g - gravitational constant

h - change in height with respect to the
pressure from the environment to the
capillary pressure

The experiment in question looks at these two equations for various screen wicks. Then, specifically, it looks at how the effective capillary radius changes for various layers of screen and how this effects the capillary pressure. It is preformed in this manner. First examine the diagram found on page 6, preceding the graphs of the data. Prior to each test the screen or screens are

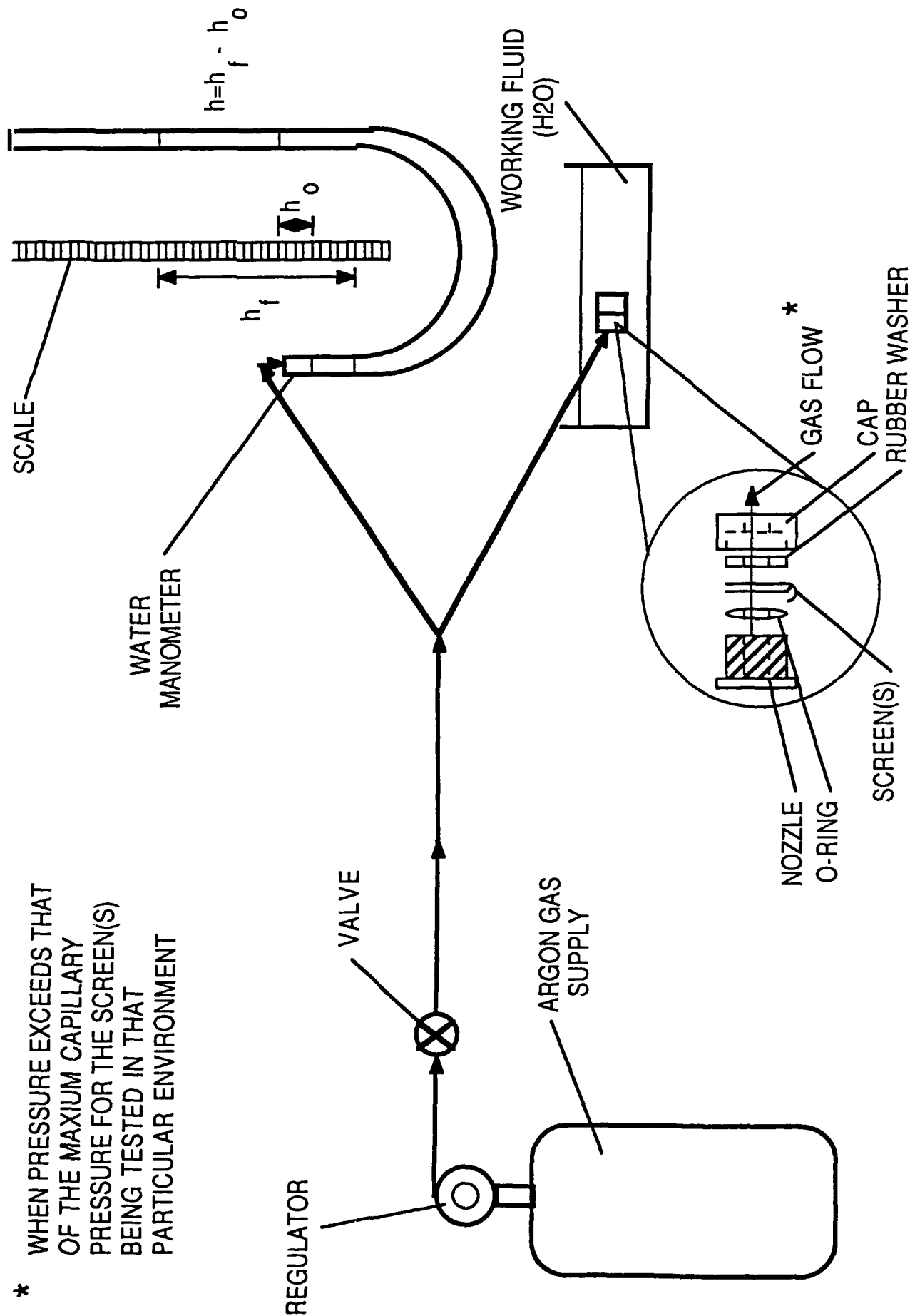
rinsed in acetone, alcohol, and soaked in oakite deoxidizer. Delimer, which is recommended for cleaning copper screens, was tried and improper data was achieved when it was used. A wire brush was used during soaking to break up any dirt. Immediately following the oakite the screens are washed in deionized water, again in alcohol, and allowed to air dry. The clean screens are then placed in between the o-ring and rubber washer in the nozzle. The nozzle is placed into the working fluid bath and a pressure difference is read and recorded from the manometer. From here the argon gas is turned onto the system until it bubbles through the screens. One must be sure that the gas is flowing through the screens and not around the washers. The rest of the system should have been leak tested prior to beginning so this is no problem. The gas can, then, be shut off. It will continue to bubble through the screens until an equilibrium is achieved. At this point, the final pressure reading is recorded from the manometer. The "h", in the equations, is figured by subtracting the initial reading from the final reading. This h value can then be used in the calculations to find a ratio between the capillary pressure and the effective capillary radius for the screens. The data has been recorded on the following graphs.

As Shown by graph #1, for 100 mesh copper screens there is no benefit for any more than 3 layers. 150 copper mesh screens can take an extra layer, 4 layers, before it does no good. Graph #2 shows that 100 mesh copper screen is more effective than 100 mesh stainless steel at any number of wraps. This fact could be true do to improper cleaning techniques, but many were tried. Graph #3

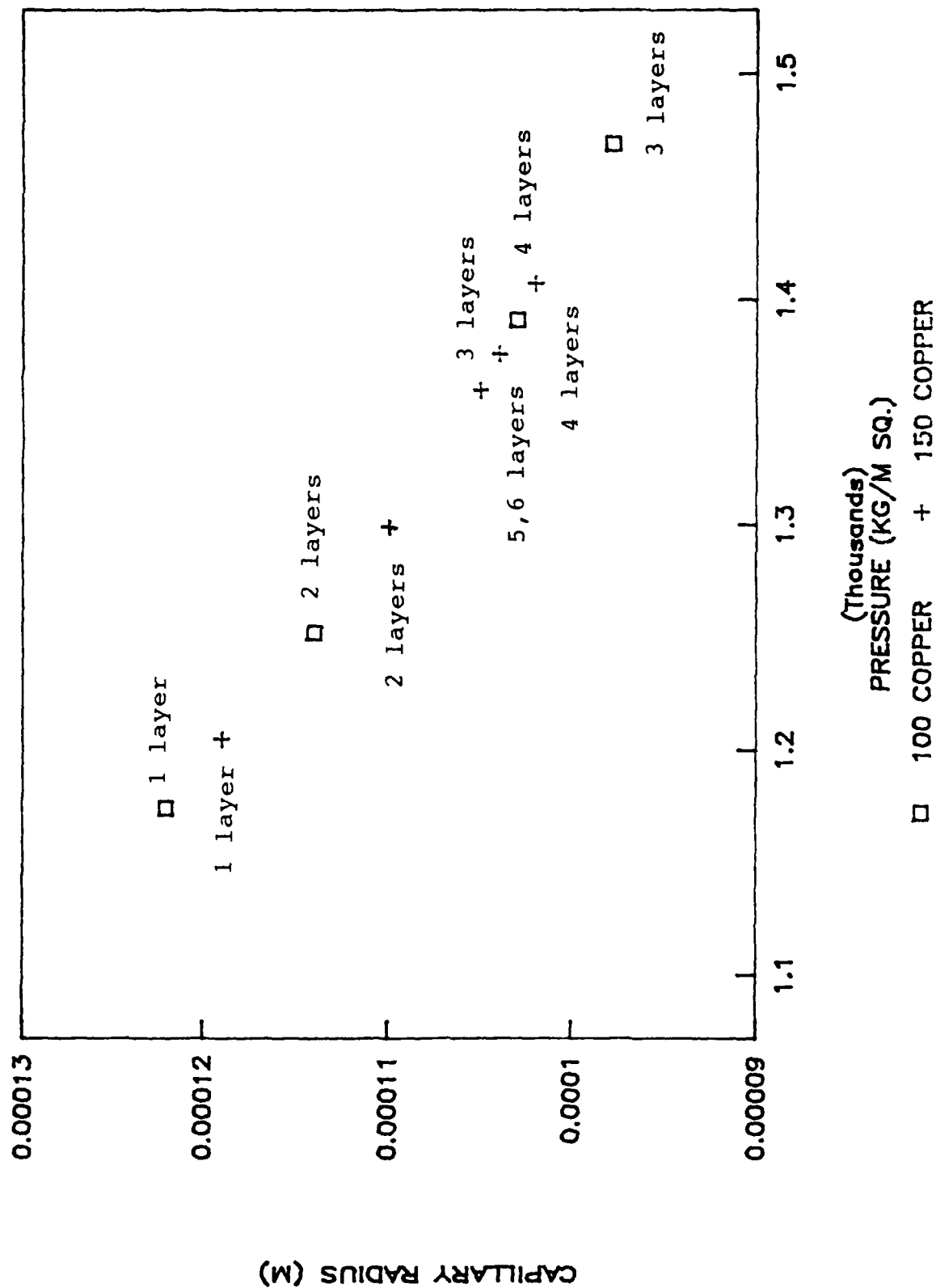
shows the difference between a screen that is cleaned properly and wets and the same screen that does not wet. When the 100 mesh stainless steel screen in graph #3 did not wet properly, a decision was made to determine the actual wetting angle. A visual observation of the angle was taken to be between 70° and 80° . When this part of the experiment was preformed, there was an attempt to gain photographic proof of the wetting angle rather than just an estimate, so that it would be possible to work from the angle and figure out the difference in pressures between the two equations. However, do to improper lighting, there was not enough light traveling through the microscope to photograph the angle. Working backwards from the change in pressure, a wetting angle of approximately 71.7° was obtained proving that the visual estimation was fairly accurate and indeed that the $\cos \theta$ term does correct the problem caused by an improper wetting angle.

CAPILLARY PRESSURE TEST SETUP

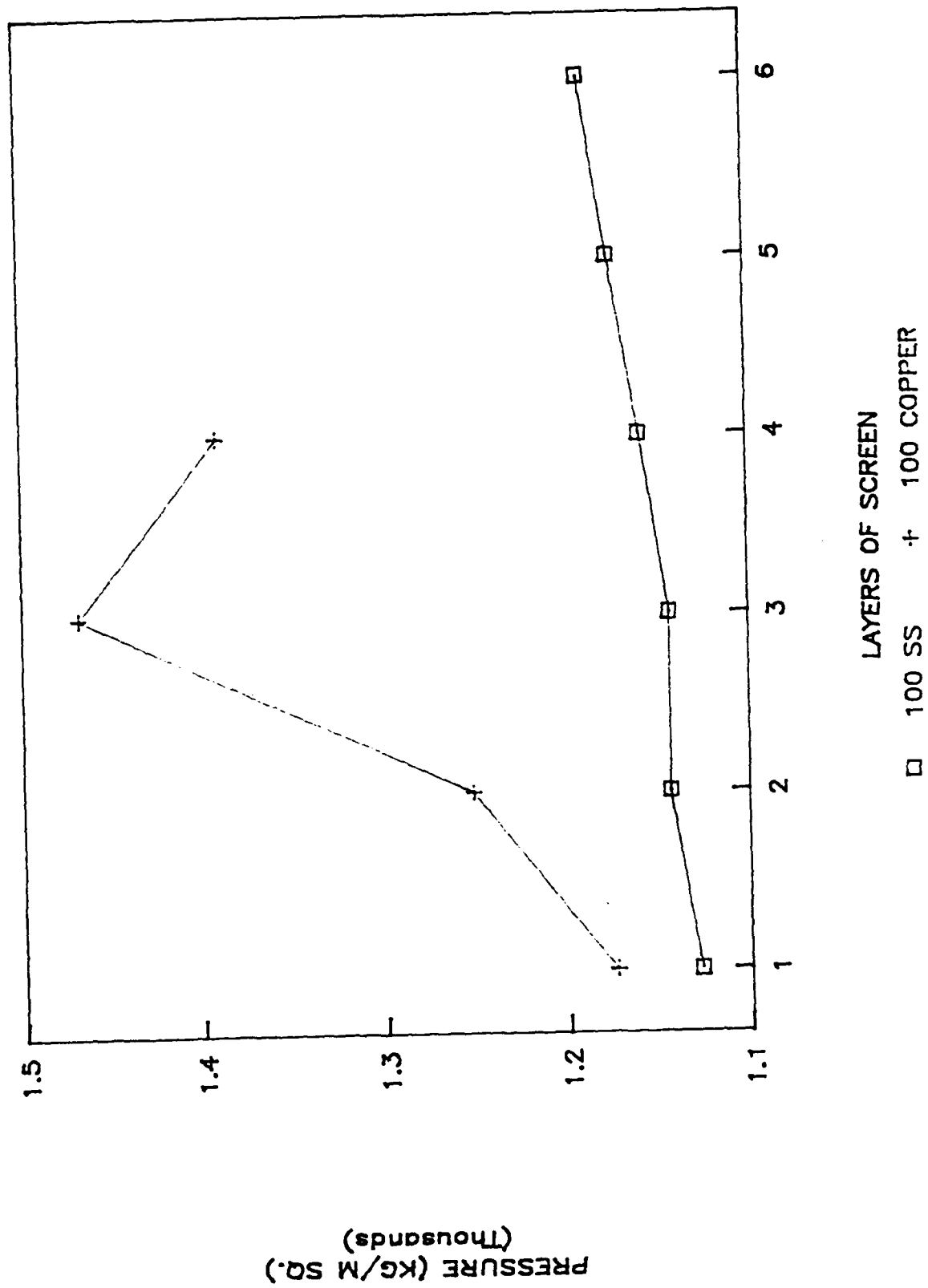
* WHEN PRESSURE EXCEEDS THAT OF THE MAXIMUM CAPILLARY PRESSURE FOR THE SCREEN(S) BEING TESTED IN THAT PARTICULAR ENVIRONMENT



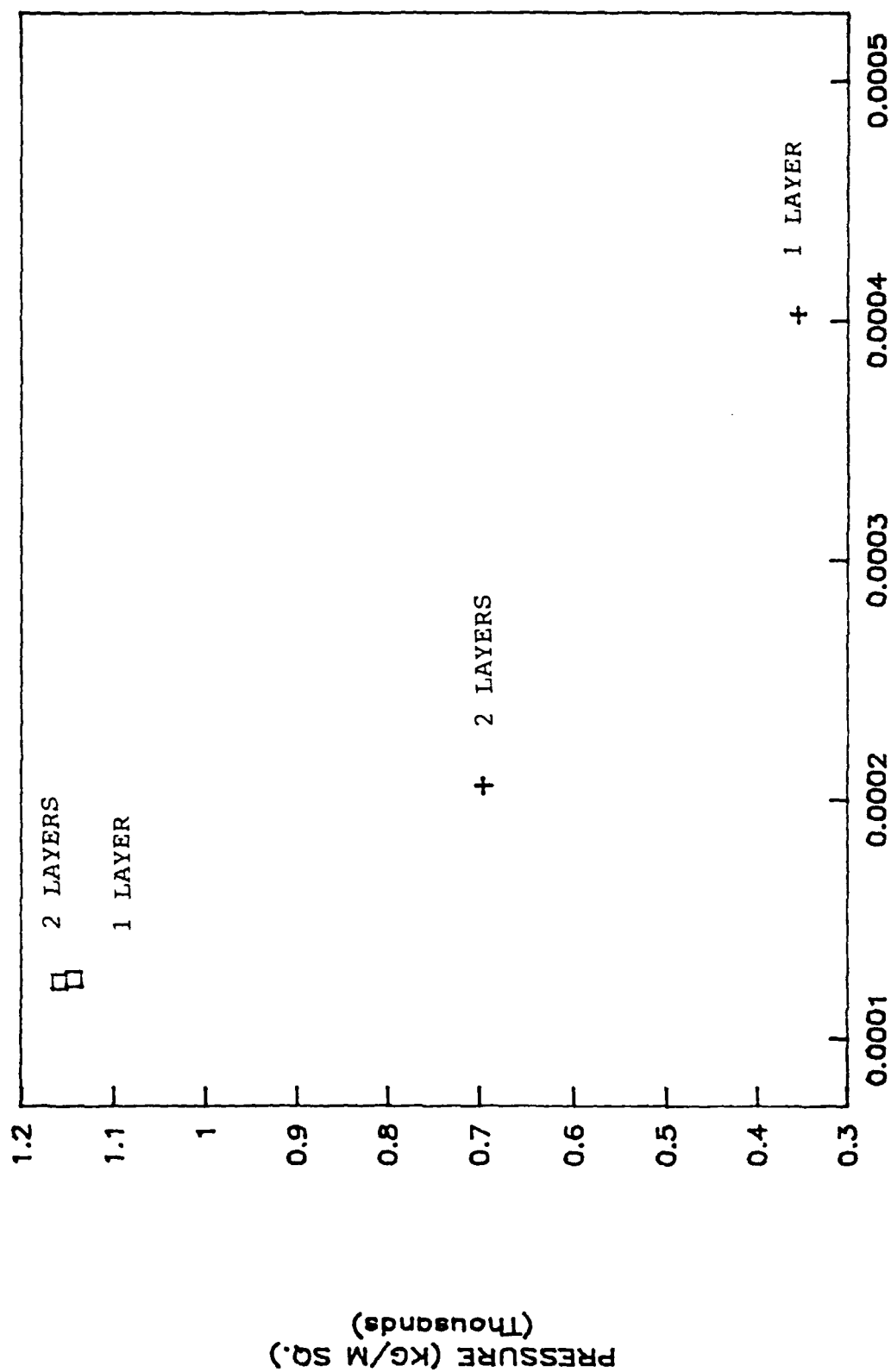
GRAPH NO. 1



GRAPH NO. 2



GRAPH NO. 3



100 STAINLESS STEEL

CAPILLARY RADIUS (M.)

□ WET SCREEN + DRY SCREEN

FINAL REPORT, SUMMER 1991

Christopher O'Dell

Throughout this past summer I learned much and did valuable work, both for myself and others. I worked mostly with computers, starting off the summer by learning the C++ programming language (one I had never worked with before). I contributed in my division by helping people with their computers, and my main project was writing a short but very complex program for a Ph.D. physicist. I also worked on smaller projects, such as the FTEPS (Fault Tolerant Electrical Power System) project. The accomplishment I am most happy with is that I used the opportunity of this employment to teach myself much and work with others.

The summer did not begin well, as my mentor, being a TAM (technical area manager), the manager of about 20 people, had much work to do and little time for an apprentice. I stayed in his office for a while, learning about the FTEPS project. Most of the research was data gathering, in effect baby-sitting a computer. The system has two generators and many safeguards against power failure in aircraft, and is quite interesting. However, for someone without a degree in electrical engineering, it is very difficult to understand, even more so to work with. Therefore, my mentor and I both agreed that I would move to a research office in a different building, the plasma physics department. Here I began work of some importance.

My first undertaking was learning the programming language C++. As

you, the reader, may know, it is a very complex, unstructured language that is difficult to learn. When I began I had some doubt as to my ability to learn it, but after a week, I was fairly proficient, and after two weeks I had written a complex, graphics based tic-tac-toe program with artificial intelligence. Then I was ready to start the task assigned to me by my new supervisor, a plasma physicist who needed some computer work done; he needed a program written in C, and I was the only one in the office who knew the language. His problem was this: he was involved with complex experimentation in which he obtained thousands and thousands of data on his computer, a Macintosh. He needed to work with those numbers on an IBM machine. The problem was that the IBM and the Macintosh store numbers differently inside the computer. Firstly, any computer stores numbers as an array of bits (a switch that is either on or off, a 1 or a 0). One of these bits is the sign bit (positive or negative), one is the exponent's sign bit, one part is the exponent on the number, and the last part, the mantissa, is the actual number. The IBM and the Macintosh store these bits in different orders, so to convert a number from one to the other one must do a lot of bit shifting and moving around of the individual bits. Although that may not sound difficult, it is. The numbers were either 32 bits (4 bytes) in length, or (even more fun) 64 bits (8 bytes) in length. I had to develop two algorithms: one to convert the 4 byte numbers, and one to convert the 8 byte numbers. It was a difficult project, and took me nearly two weeks to complete (a copy of the two programs is attached at the end of this report, appendices A and B).

My boss and I learned much about the way the two computers store

numbers after I finished the project. We learned that the IBM has three more significant bits in the mantissa, and that the Macintosh, for some unknown reason, always adds 1 to the exponent (the people at Macintosh thought it was two - we corrected them). The finished programs were then employed into larger programs that worked with the data from my boss's experiments. It worked well, and I was proud that it was stunningly fast and relatively simple to understand. After the completion of the project, I spent the rest of the summer mastering C++, and wrote a mathematical graphics package that plots equations, finds points of intersection, etc. I also helped people here with their computers, installing programs for them, backing up their disks, etc.

Over all, this summer was very successful. Next year at college I will be able to put to use much of what I learned this summer, as I begin my major in physics and computer science. I have a new programming language under my belt, along with practical experience working with people. I am happy that I completed a project of vital importance, if only for a few people involved in that program. I am very pleased with my accomplishments this summer, and I am sure I have enhanced my education and my life by being a part of this program.

APPENDIX A

```
/* PROGRAMMER : Chris O'Dell

   COMPLETED : July 30, 1991

   PURPOSE    : This function is designed to receive one floating
                 point number, 4 bytes in length, in the macintosh
                 style of writing numbers. It will then convert it
                 to the style that the ibm writes it in, and will
                 return that value.

   ** NOTE     : 1 BYTE CONTAINS 2 HEXADECIMAL DIGITS
                 1 BYTE ALSO CONTAINS 8 BITS

*/

float convert(float macnum)
{
/* VARIABLES */

    union
    {
        float number;          /* number AND byte[4] HOLD
        unsigned char byte[4]   SAME MEMORY SPACE          */
    }
    u;
    unsigned char temp;        /* VARIABLE 1 BYTE IN LENGTH    */
    int count;

/* START CODE */

    u.number = macnum;          /* PUT MACNUM INTO UNION        */
    for (count=0;count<=1;count++)
    {
        temp = u.byte[count];
        u.byte[count]=u.byte[count+2]; /* SWITCH FIRST 2 BYTES
        u.byte[count+2]=temp;          WITH LAST 2 BYTES          */
    }
    if (u.byte[3]!=0) u.byte[3]--; /* SUBTRACT 1 FROM FIRST BYTE */
    return(u.number);

/* END CODE */
}
```

APPENDIX B

/* PROGRAMMER : Chris O'Dell

COMPLETED : July 30, 1991

PURPOSE : This function is designed to receive one floating point number, 8 bytes in length, in the macintosh style of writing numbers. It will then convert it to the style that the ibm writes it in, and will return that value.

**NOTE : 1 BYTE CONTAINS 2 HEXADECIMAL DIGITS
1 BYTE ALSO CONTAINS 8 BITS

*/

```
double convert(double macnum)
{
```

```
/* VARIABLES */
```

```
union
```

```
{
```

```
    double number;
```

```
    unsigned char byte[8];
```

```
}
```

```
u ;
```

```
unsigned char mask,mask2,temp;    /* ONE BYTE NUMBERS */
```

```
int count;
```

```
/* START CODE */
```

```
u.number = macnum;    /* PUT NUMBER INTO UNION */
```

```
for (count=0;count<=6;count+=2)    /* RESHUFFLE BYTES */
```

```
{
```

```
    temp = u.byte[count];
```

```
    u.byte[count]=u.byte[count+1];
```

```
    u.byte[count+1]=temp;
```

```
}
```

```
if (u.byte[0]!=0) u.byte[0]-=1;    /* SUBTRACT 1 FROM FIRST BYTE */
```

```
mask = 7;    /* 00000111 IN BINARY; ISOLATES  
                LAST 3 BITS IN A BYTE    */
```

```
for (count=7;count>=1;count--)    /* RIGHT - SHIFT ENTIRE ARRAY 3 */  
{    /* (EXCEPT FIRST BYTE)    */
```

```
    u.byte[count] = (u.byte[count] >> 3) | ((mask & u.byte[count-1])*32);
```

```

}

/* MESS WITH FIRST BYTE: (8 BITS, 0 THROUGH 7)
THE FOLLOWING CODE:
-----
1) MOVES BITS 2,3,& 4 INTO POSITIONS 5,6,& 7.
2) IF BIT 1 IS A 0, MAKES BIT 4 A 1, ELSE MAKES
   BIT 4 A 0.
3) DEFINES BITS 0 THROUGH 3 ACCORDING TO THE SIGN BITS (0 & 1)
   00 --> 3 (0011)
   01 --> 4 (0100)
   10 --> B (1011)
   11 --> C (1100)
*/

mask = 192 & u.byte[0];                /* FIRST TWO BITS (SIGN BITS) */
u.byte[0] = (u.byte[0] >> 3) ;
mask2 = 8 & u.byte[0];                  /* EXPONENT BIT */
if (mask2==0) u.byte[0] != 8;
    else u.byte[0] &= 247;
u.byte[0] &= 15;                        /* MAKE FIRST 4 BITS ZEROS */
if (mask==0) u.byte[0] != 48;            /* MAKE FIRST DIGIT A '3' */
    else if (mask==64) u.byte[0] != 64;  /* MAKE FIRST DIGIT A '4' */
    else if (mask==128) u.byte[0] != 176; /* MAKE FIRST DIGIT A 'B' */
    else u.byte[0] != 192;               /* MAKE FIRST DIGIT A 'C' */

for (count=0;count<=3;count++)          /* REVERSE BYTES */
{
    temp = u.byte[count];
    u.byte[count]=u.byte[7-count];
    u.byte[7-count]=temp;
}
if (macnum==0) u.number=0;              /* SELF-EXPLANATORY */
return(u.number);

/* END CODE */
}

```

FINAL REPORT TO RDL
COMPRESSOR RESEARCH FACILITY TEST GROUP

JENNIFER L. POLLOCK

This summer I was placed in the Aero Propulsion and Power Laboratory, assigned to the Compressor Research Facility Test Group. My mentor and the Technical Area Manager of the Test Group was Norman D. Poti. The assignment I was given for my summer research included producing a form of plot called a Campbell Diagram, forming tables from these diagrams, and analyzing the diagrams.

In the course of my summer research, my main objective was to learn the function of the Test Group in the Compressor Research Facility. The facility is part of the Air Force Aero Propulsion and Power Directorate, and is used to evaluate the performance of gas turbine engines. The CRF has the capability to test almost any compressor configuration with confidence in the and reliability of all data taken during the test. This confidence was achieved through the use of a fully automated and computer controlled facility, and the ability to take data of over 700 individual points or areas in the compressor. To help me learn what the Test Group does, I was taught how to use a software program written by Battelle Corporation for the Compressor Research Facility. This program was used to convert analog tape data into digital data and reorganize the data so that it could be used for the Campbell Diagrams. This analog data is taken

during the test runs. The data I used were of three different measurements. They were of the strain on specific areas in the compressor, the frequencies at which the strain occurred, and the magnitude of the strain. The Campbell Diagrams display this data as an XY plot where the RPM at which the compressor is turning is plotted against the frequency of the strain, and the magnitude of the strain is shown is KSI peak to peak as the length of the bars on the graph (Figure 1).

The process of digitizing the data involved several steps (Figure 2). The first step was to get the specific tape that held the data I needed to digitize. I found the tapes by reviewing the notes taken by the engineers present during the test run. After locating the tape, I loaded it onto a tape deck and viewed the data on oscilloscopes in the facility. I then selected channels from the tape that appeared to contain "good" data (Figure 3a & b), that is information that was actually present, not noise or interference (Figure 4a & b). After selecting the channels, I used the Battelle program to digitize the data and produce the Campbell Diagrams (Figure 5). The Battelle program also created Campbell tables (Figure 6) which I used in my research. The tables showed the larger magnitudes of strain and the order line on which they occurred. This data was used to make summaries of the runs. The purpose of the summaries (Figure 7) was to compare the actual strain on the various parts of the compressor to the strain limits. The

manufacturer sets the strain limits for the engine. The purpose of the limits is to prevent damage to the engine. The limits are also used to determine if the engine needs further modifications or redesign, or if the engine is acceptable in its present form.

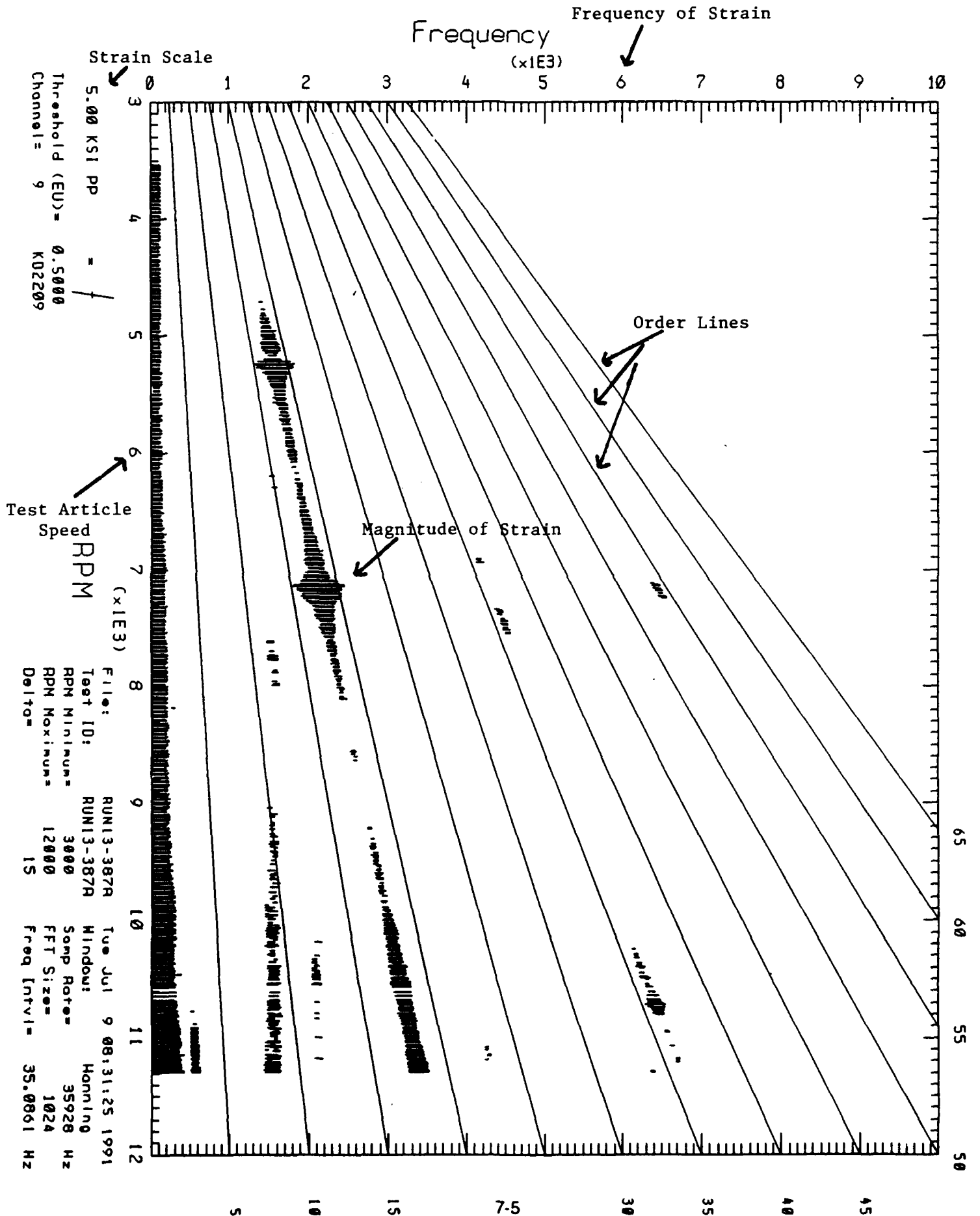
Another of the projects I worked on also involved the use of analog data, but for this project the data were not digitized. My task for this assignment was to take data directly from the analog tapes and plot them on an XY plotter. These data were of the actual vibrations that are associated with the rotating rotor shaft of the compressor. The plots showed that throughout the test there was very little decomposition in the rotor, and the decomposition that did occur was within the acceptable limits set up by the manufacturer (Figure 8a & 8b).

In the beginning of my apprenticeship the data that I used were from a previous test done in the Compressor Research Facility. A new test was scheduled to begin while I was working at the Facility. For this test I was involved in the pretest setup and facility checks. One of the tasks that I helped complete was a check called "end to ends", this is actually a calibration done to ensure that the signals coming from the sensors on the test article show up the same on the oscilloscopes, the computer terminals, and on the recording tapes. Another of the tasks that I did was to prepare the oscilloscopes for each evenings' run. I took readings from each oscilloscope and listed the readings in

notes specifically for that purpose. After completing the night's run, one of the engineers present (I could not be present because I lacked a security clearance.) took new readings to compare with the ones I took. This was done to make sure that the oscilloscopes were still calibrated correctly. In the daily preparation I also had the task of preparing some of the other equipment for the run. The equipment included a computer system called the MTI, several spectrum analyzers, printers, and the tape decks.

During my apprenticeship in the Compressor Research Facility Test Group I was given the opportunity to work in the field that I plan to major in in college. While at the facility I had the chance to use equipment that I will use in my future career, learn how teamwork leads to a successful working environment, learn about responsibility to myself and my co-workers, and learn about the operation of the Compressor Research Facility Test Group.

FIGURE 1



OVERVIEW OF DIGITIZING STEPS

REVIEW ENGINEERS'S NOTES

LOAD ANALOG TAPE ON TAPE DECK

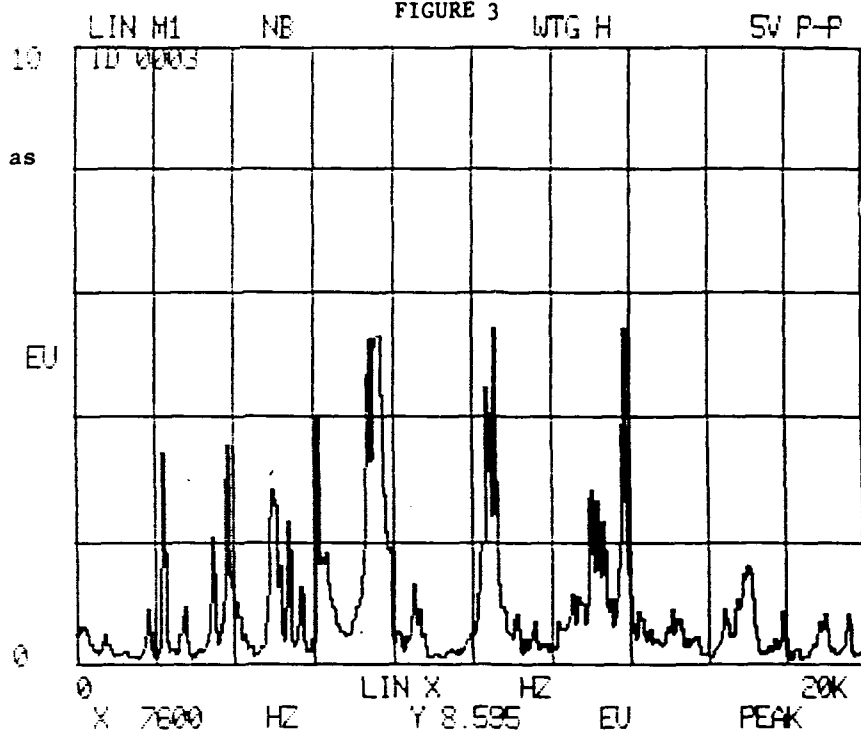
VIEW DATA ON OSCILLOSCOPES

SELECT CHANNELS

USE BATTELLE DIGITIZING PROGRAM
ON MASSCOMP COMPUTER SYSTEM

MAKE CAMPBELL DIAGRAMS,
TABLES, AND RUN SUMMARIES

FIGURE 3



a) "Good" data as shown on a spectrum analyzer.

b) "Good" data in the form of a Campbell Diagram.

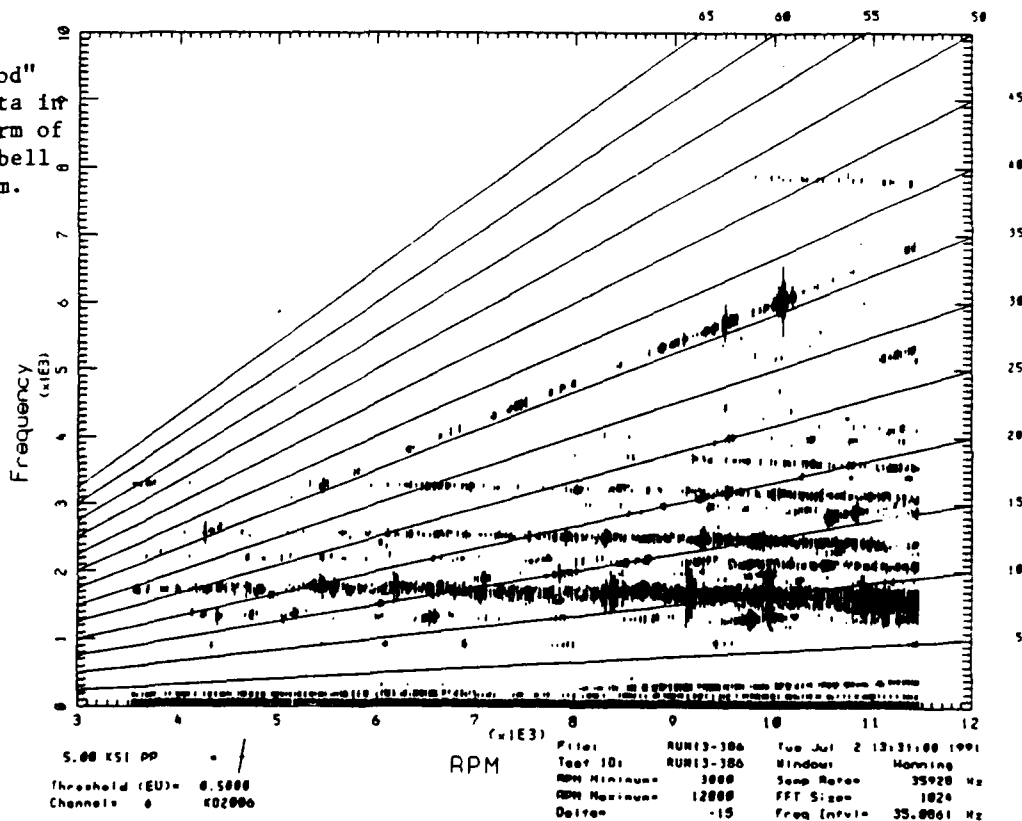
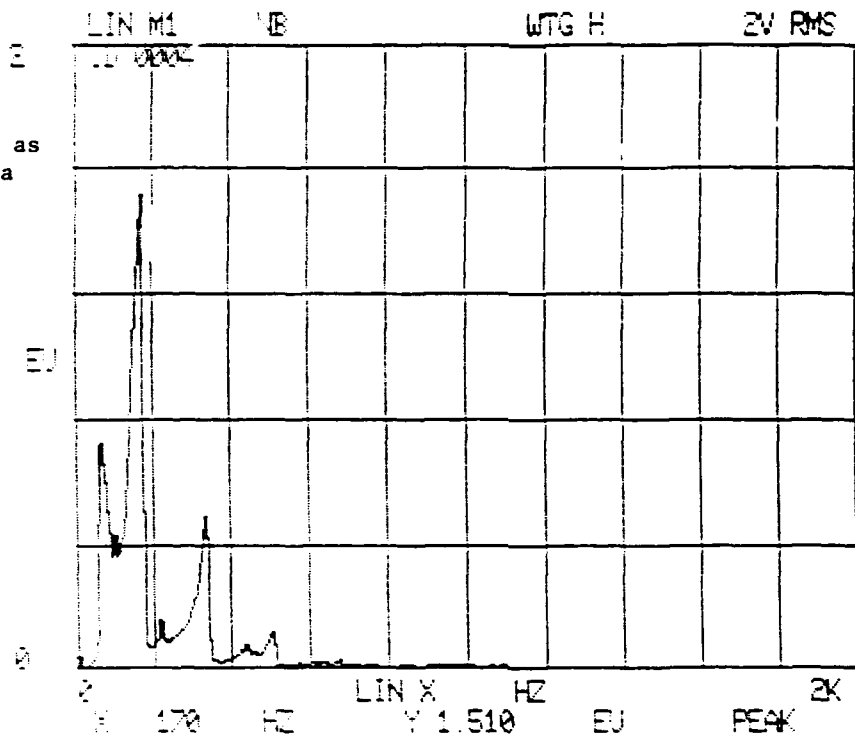


FIGURE 4

a) "Bad" data as shown on a spectrum analyzer.



b) "Bad" data in the form of a Campbell Diagram.

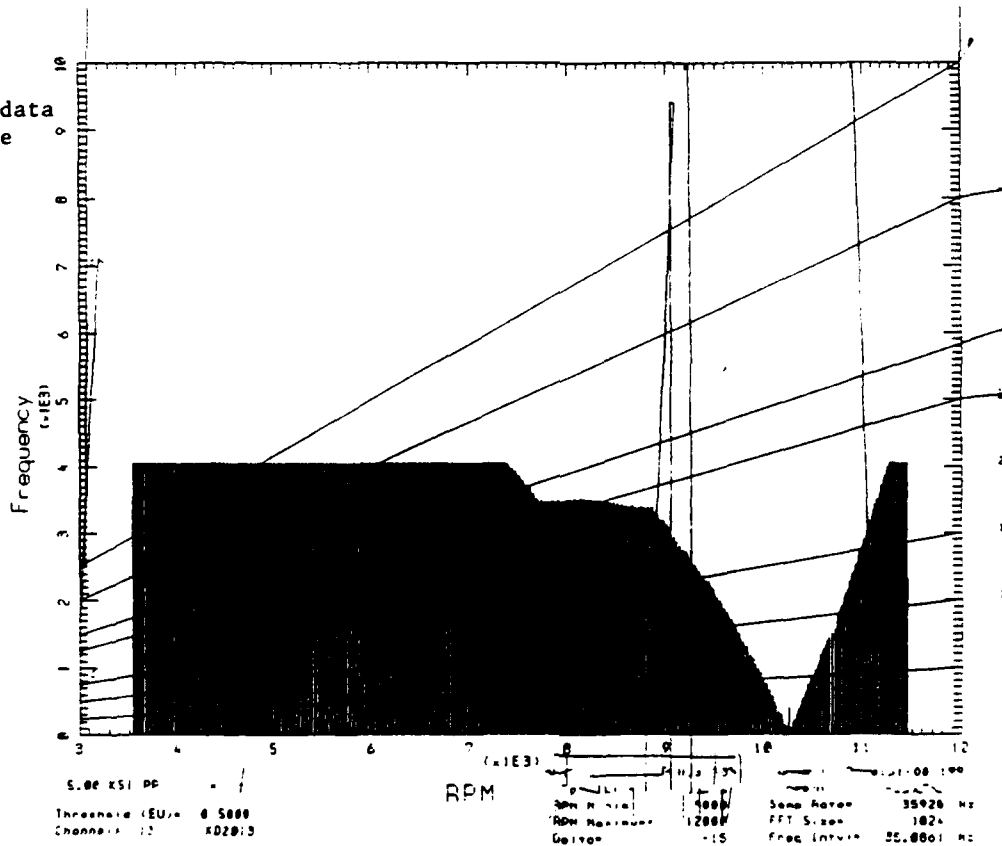


FIGURE 5

Example Campbell Diagram

Frequency

($\times 1E3$)

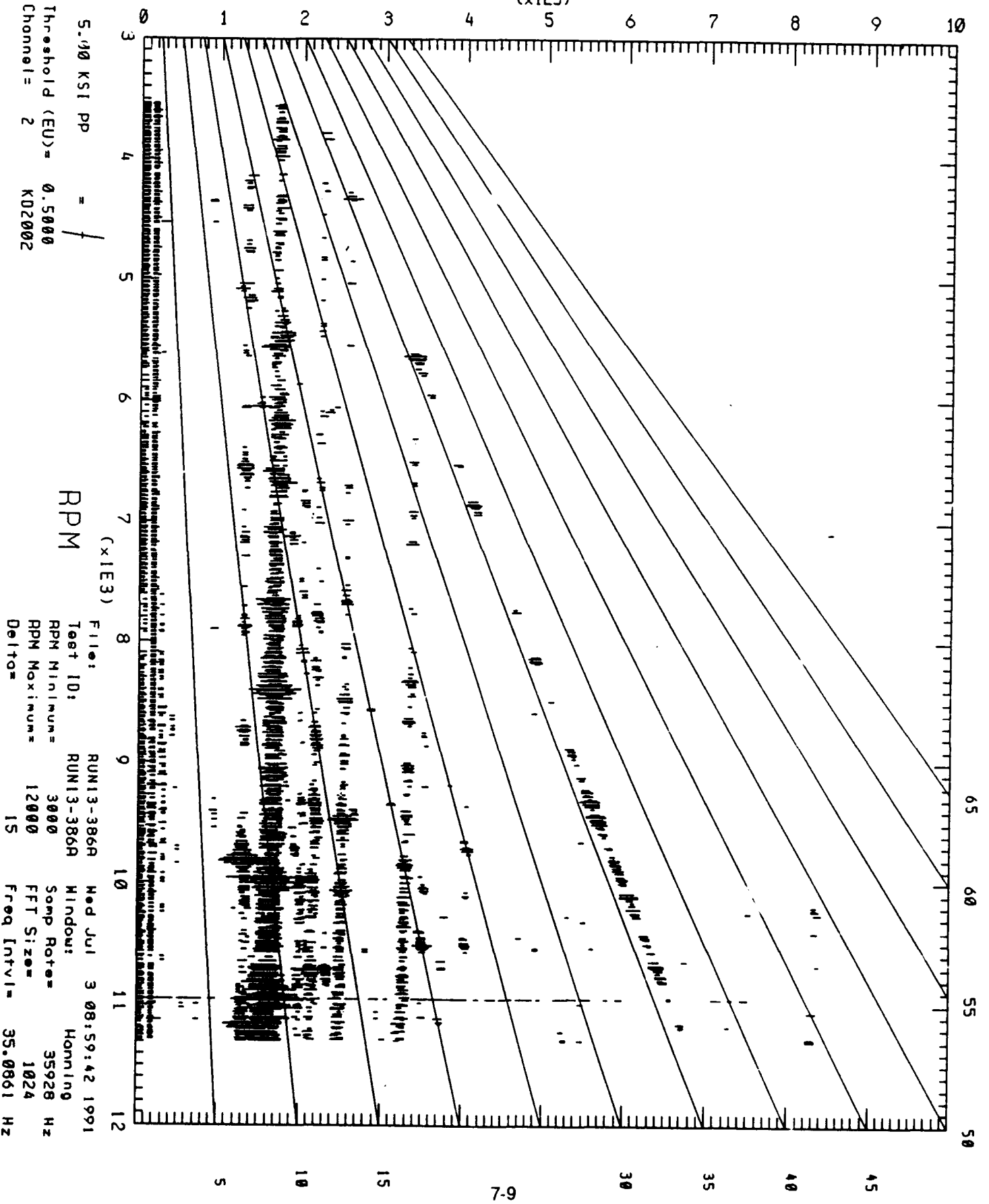


FIGURE 6

Camobell Diagram Data Table

FFT-Base Creation Date: Mon Aug 5 08:00:14 1991

Test ID: F119B-RUN3-B

FFT Database: F119B-RUN3-B

FFT Order:	10	Resultant Freq. Interval:	39.06
Minimum Plotted Freq.:	0.00	Maximum Plotted Freq.:	15000.00
Minimum RPM in Dataset:	3604.00	Maximum RPM in Dataset:	10032.00
Minimum Plotted RPM:	3000.00	Maximum Plotted RPM:	10000.00
Delta of RPM:	15.00	Sampling Rate:	40000.00
Threshold:	5.00	Window:	Hanning
Channel:	9		
Plot title:	SLOW ACCEL ON DOL		

FFT	RPM	Freq/Hz	Mag/KSI PP	Order Line
232	7132.00	7617.19	5.23	64.08
233	7146.00	7617.19	7.36	63.96
234	7162.00	7617.19	5.12	63.81
234	7162.00	7656.25	6.07	64.14
235	7180.00	7656.25	6.25	63.98
236	7192.00	7656.25	8.42	63.87
236	7192.00	7695.31	6.91	64.20
237	7208.00	7656.25	7.33	63.73
237	7208.00	7695.31	10.66	64.06
238	7222.00	7695.31	12.12	63.93
238	7222.00	7734.38	8.29	64.26
239	7240.00	7695.31	11.23	63.77
239	7240.00	7734.38	12.94	64.10
240	7252.00	7695.31	7.40	63.67
240	7252.00	7734.38	15.99	63.99
240	7252.00	7773.44	9.03	64.31
241	7270.00	7734.38	16.67	63.83
241	7270.00	7773.44	16.28	64.15
242	7284.00	7734.38	9.00	63.71
242	7284.00	7773.44	14.81	64.03
242	7284.00	7812.50	6.21	64.35
243	7304.00	7773.44	6.75	63.86
243	7304.00	7812.50	5.41	64.18
244	7318.00	7812.50	6.84	64.05

FIGURE 7

Example Summary

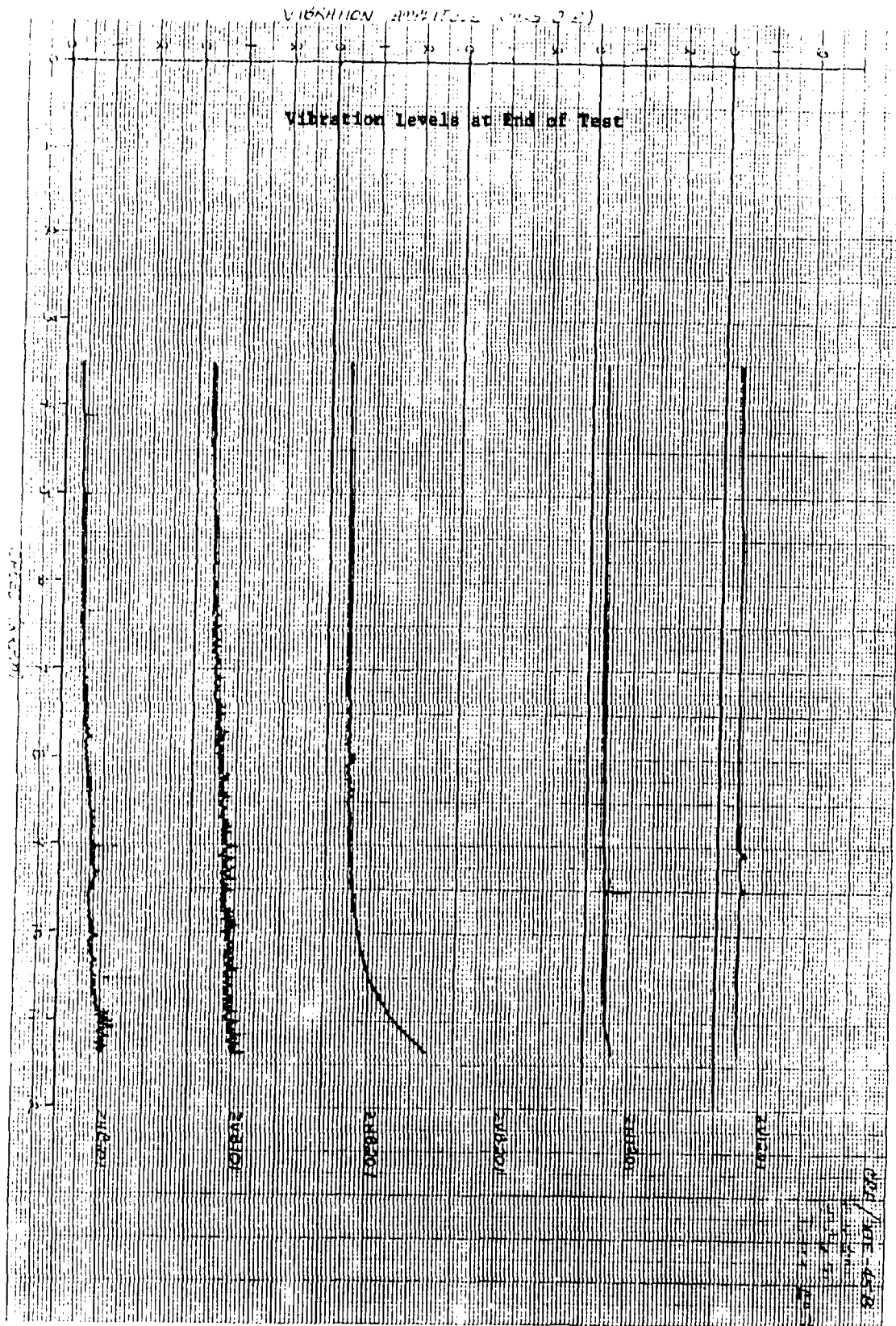
XTE-45B FAN RIG STRESS SUMMARY
 RUN 1 (2/5/91) SLOW DECEL (15 RPM/SEC)
 HOLLOW ROTOR, CIRCUMFERENTIALLY GROOVED TIP TREATMENT

BASED ON CAMPBELL DIAGRAMS

STAGE	GAGE HEADER	SPEED (RPM)	FREQ (HZ)	MODE	N/REV	STRESS (KSI)	LIMIT (KSI)	% LIMITS
FAN ROTOR	KD2005	9338	5614	UNK	36	3.9	51	8
	KD2006	9589	491	1F	3	3.1	14	22
	KD2007	11084	2210	CMPLX	12	4.3	5	68
		10857	1614	CMPLX	9	5.5	38	15
		9868	1649	CMPLX	10	4.6	38	12
		9003	2246	CMPLX	15	5.9	36	16
	KD2008	10255	1368	1T	8	3.2	20	16
		9659	5789	UNK	36	3.2	28	11
	KD2009	9589	491	1F	3	2.1	10	21
	KD2010	9324	5579	UNK	36	2.8	61	5
	KD2011	9236	456	1F	3	3.7	14	26
	KD2012	10369	6210	UNK	36	6.1	39	16
		10061	3368	---	20	4.6	---	---
		10045	2667	CMPLX	16	4.3	35	12
	KD2014	9514	947	2F	6	1.8	20	9
		9236	4546	---	3	2.2	---	---
CORE IGV	KD2501	11198	1123	1F	6	7.2	34.6	21
	KD2505	11217	1123	1F	6	22.5	28.2	80
	KD2507	11217	1123	1F	6	16.9	28.2	60
	KD2508	11053	1088	1F	6	19.9	33.9	59
FAN OGV	KD2201	7126	2140	1T	18	4	9.3	43
	KD2202	7155	2140	1T	18	7	21.7	32
	KD2205	11134	1474	1F	8	3.4	11.7	29
		7210	2175	1T	18	9.7	14.3	68
	KD2207	7136	2140	1T	18	8.3	14.3	58
	KD2208	7149	2140	1T	18	8.2	9.3	88

1998





Space Based Interceptor Star Tracker Analysis

by High School Apprentice: Eric Apfel

Mentor: Tim Poth

Department: MNSI

Date: 12 August 1991

SPACE BASED INTERCEPTOR STAR TRACKER ANALYSIS

High School Apprentice: Eric Apfel

I. INTRODUCTION

As a high school apprentice working to learn the various tools and skills required in the fields of science and engineering, most of my gained knowledge came from writing computer programs. These programs were designed to perform functions ranging from plotting the history of a gyro trajectory over a certain interval of time to determining how many stars a given orbiting telescope can no longer see due to various sources of noise. Writing each program and sorting out its bugs was a task that varied in time according to complexity (some programs took me under a day, while others lasted me over a week!). I feel proud of the work I achieved, however, and I will discuss in detail each of the programs (excluding the ones which were simply fruitless attempts to improve accuracy) after an explanation as to why I wrote these programs.

I work in the Munitions Directorate's Guided Interceptor Technology Branch, abbreviated MNSI. The specific section I work in involves guidance and navigation of space based interceptors. These interceptors are designed to orbit the planet and could be activated in response to the detection of hostile ICBM's. The interceptors could then be launched to intercept and destroy the ballistic missiles. No explosive warhead is involved, but instead the kinetic energy of the interceptor is used to kill the missile. Accuracy, therefore, is of high importance; and proximity hits, which were acceptable with explosive warheads, will be useless with the interceptors. Since these interceptors will be orbiting the planet for long periods of time (around 5 to 10 years) it is necessary to be very accurate in knowing their attitude and heading at each and every moment during their time of operation.

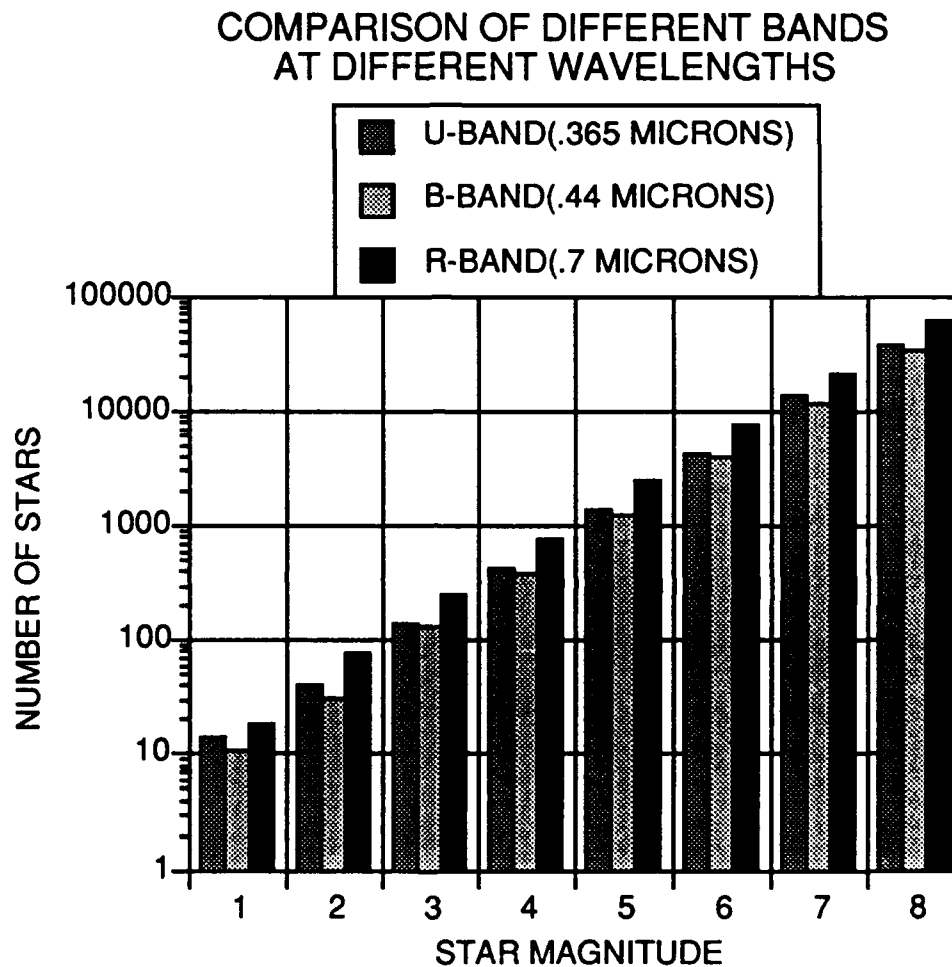
My first program plotted the history of an interceptor's attitude as determined by small mechanical devices called gyros. One gyro being developed at MNSI is called MIGS (micromechanical inertial guidance system). Three of these devices are to be installed on an interceptor to determine changes in roll, pitch, and yaw. The program plots the trajectory based on thrust intervals in the three spatial directions and includes various errors due to bias and Brownian motion (these devices are microscopic in size and easily affected by air molecules!). The program results in a fairly accurate description of the history of the interceptor's trajectory, but even very minute errors in interceptor positioning and attitude can result in gross deviations as the time progresses (a prime example of chaos theory). It is therefore not sufficient to merely give the interceptor occasional attitude adjustments (no pun intended). Instead, to further enhance the accuracy of the interceptor's attitude, another device is in the planning stages at MNSI. This project is nicknamed Mini-OWLS (short for miniature optical wide-angle lens star tracker). It includes three very small telescopes which can determine the interceptor's attitude by viewing the stars. Thus we can rely on the stars to aid in navigation just as the historical seafaring explorers. Star detection, however, is a problem of its own.

II. Star Availability/Detectability Analysis Tool Development

I wrote many programs to achieve the final goal of determining the best method of star detection. The first of which involved reducing a very large data file of catalogued stars to a size which could easily be managed and manipulated. (Actually two reduced data files were created. One contained only the most essential information to be used and was approximately five percent the size of its parent. The other contained information which was used infrequently and measured about 20 percent of the catalogue's size.) A series of programs were then written to develop charts which could be used to determine which waveband would be best to view the stars and which points of the telescope's orbit would result in a

shortage of stars (Figs. 1,2,&3). To determine the number of stars detectable in each waveband, it was necessary to know the adjusted magnitude of the star in the new waveband.

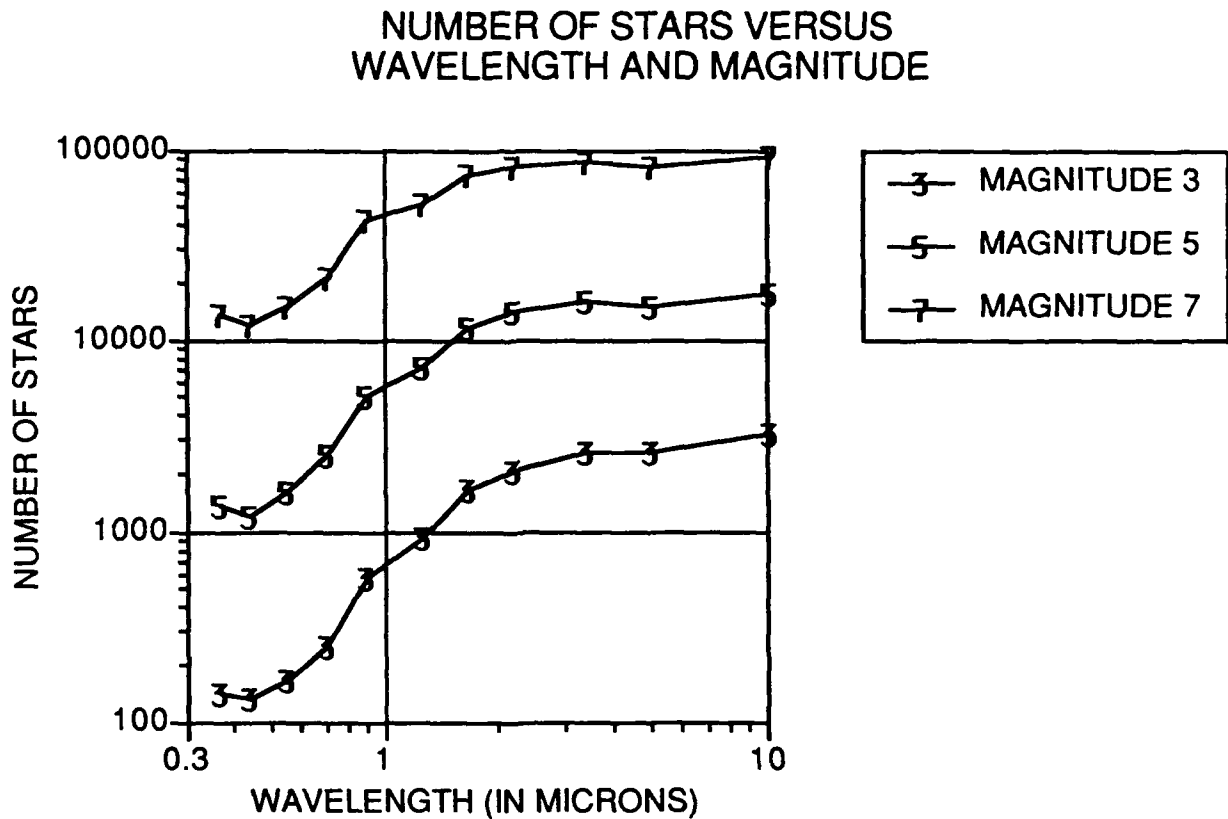
Figure 1



The R-band, U-band, and B-band magnitudes were calculated from the visual magnitude based on a series of formulas which shall be presented here in steps:

Step 1: Determine the temperature of the star from its spectral class with the aid of a lookup table.

Figure 2

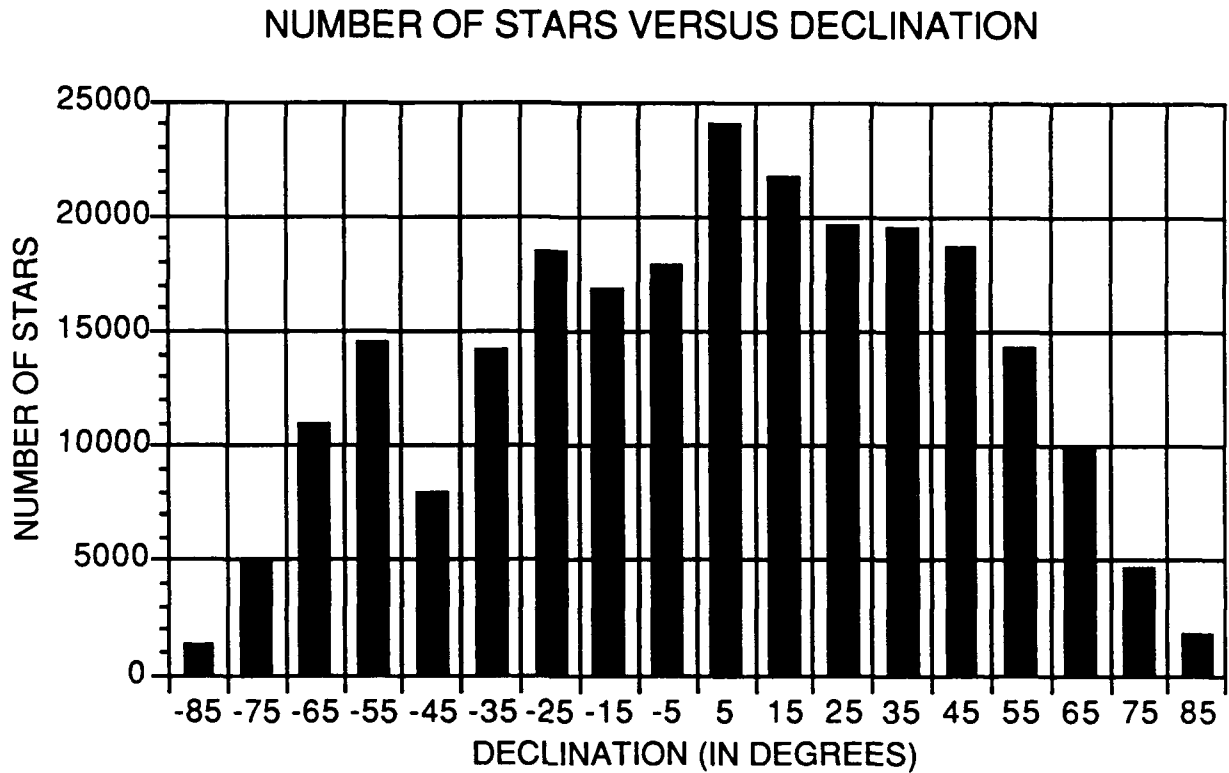


Step 2: Compute the spectral irradiance of a zero magnitude star (Vega). This value is computed using a formula which was derived by myself in an attempt to increase the accuracy of the programs (the original procedure required the aid of a lookup table obtained from Rockwell's Surveillance, Tracking and Ground Systems group). The formula is displayed in equation 1

$$H_{0_{\text{peak}}} = (6.28263)10^{-33}(e^{14387.9/.55T}-1)T^5 \quad (1)$$

where T is the temperature of the star in °K while e is the constant 2.718... $H_{0_{\text{peak}}}$ is the spectral irradiance of a zero visual magnitude star at the specified temperature. The

Figure 3



derivation of this equation will be listed in the appendix for those who are interested.

Step 3: Compute peak spectral irradiance. This can be done with the formula:

$$H_{\text{peak}} = H_{0\text{peak}}(10^{-m_v/2.5}) \quad (2)$$

where m_v is the stars visual magnitude, and H_{peak} is its peak spectral irradiance. This formula is derived from the quantitative relation between magnitudes and fluxes for celestial sources, i.e. $m_B - m_A = 2.5 \log(\frac{S_A}{S_B})$. Letting $m_B = 0$

and $S_B = H_{0_{peak}}$, we can get Equation 2.

Step 4: Compute the spectral irradiance at the appropriate wavelength using the formula:

$$H_{(l)} = H_{peak} * \frac{Q^5}{21.199(e^Q - 1)} \quad (3)$$

where H_{peak} is the peak spectral irradiance, e is the constant 2.7182818..., $H_{(l)}$ is the estimated spectral irradiance at wavelength l , and Q is a constant calculated with the formula:

$$Q = \frac{14387.9}{(lT)} \quad (4)$$

where T is the temperature of the star in ∞K and l is the wavelength in microns.

Step 5: Compute the magnitude in the adjusted waveband. This can be done with the formula:

$$m_s = -2.5 \log\left(\frac{H_{(l)}}{H_0}\right) \quad (5)$$

in which H_0 is the spectral irradiance of a zero magnitude star in the waveband of interest (determined by a table), and m_s is the magnitude in the wavelength of interest.

My program handles the above algorithm easily and quickly (it performs this algorithm for all 241,592 stars in under two minutes on the Cray supercomputer) and the results appear accurate. The next step after calculating adjusted magnitudes is to determine the number of stars which can be detected for each waveband. In order to accomplish this task, the signal to noise ratio of each star must be computed and compared to a required ratio. If the calculated signal

to noise ratio is high enough, then theoretically we can detect it. The steps involved in calculating the signal to noise ratio are outlined below.

Step 1: Compute the peak spectral irradiance (H_{peak}).
(Equation 2)

Step 2: Compute the spectral irradiance over all wavelengths with the formula:

$$SR = Z\sigma T^4 \quad (6)$$

where SR is the total spectral irradiance, σ is the Stefan-Boltzman constant (5.669×10^{-12}) (Watts/cm²K⁵), T is the star's temperature in °K, and Z is calculated using the formula:

$$Z = H_{\text{peak}} * \frac{7.78 \times 10^{14}}{T^5} \quad (7)$$

in which T is once again temperature in °K, and H_{peak} is the peak spectral irradiance ($\frac{\text{Watts}}{\text{cm}^2\mu}$).

Step 3: Compute bandlimited spectral irradiance. This process involves determining spectral irradiance between zero wavelength and the extremes of the bandwidth and using a lookup table. By subtracting the fractional energy at the lower wavelength from the fractional energy at the higher wavelength, the energy difference can be determined as shown in Equation 8.

$$E_{\lambda_1-\lambda_2} = E_{0-\lambda_2} - E_{0-\lambda_1} \quad (8)$$

This fractional energy is then multiplied by the spectral irradiance over all wavelengths to arrive at the bandlimited

spectral irradiance (Equation 9).

$$\text{Trad} = E_{\lambda_1-\lambda_2}SR \quad (9)$$

Step 4: Compute the photon flux from the star with the formula:

$$\text{PF} = \frac{\text{Trad}\lambda}{hc} \quad \frac{\text{Photons/sec}}{\text{cm}^2\mu} \quad (10)$$

where Trad is the bandlimited spectral irradiance from Step 3, λ is the wavelength, h is Planck's constant (Joule seconds), c is the speed of light (in $\frac{\mu}{s}$), and PF is the photon flux.

Step 5: Compute the detected signal in photons due to the star's energy using:

$$S = \frac{(\text{PF})(A)(\text{NT})(\text{QE})(\text{TS})}{\text{NS}} \quad (11)$$

in which PF is the photon flux, A is the aperture area (in cm^2), NT is the transmission efficiency, QE is the quantum efficiency of the detectors, TS is the signal integration time (in seconds), and NS is the number of pixels over which the blur spot is detected.

Step 5: Compute the detector noise in electrons using the formula:

$$\text{ND} = \sqrt{S + (\text{DC})(\text{TS})} \quad (12)$$

In this formula, S is the detected signal, DC is the dark current, TS is the signal integration time, and ND is the detector noise.

Step 6: Compute signal to noise ratio:

$$\text{SNR} = \frac{S}{\text{ND}} \quad (13)$$

where S is the detected signal, ND is the detector noise in electrons, and SNR is the signal to noise ratio.

The signal to noise ratio can then be compared to the required signal to noise ratio to determine the number of stars that can be detected by a specific startracker.

III. Startracker Design and Analysis

The completion of this program prompted the immediate application of sensitivity studies. A set of baseline parameters were created (Table 1) from which one parameter was varied at a time to determine the effect on the number of stars detected. The waveband center, aperture area, signal integration time, pixel spread number, dark current, and required signal to noise ratio were all varied to determine the sensitivity of each parameter. The task was aided by the use of a separate program to create files consisting of the desired

Table 1.

Waveband center wavelength	.7 μ
Waveband	.6 - .8 μ
Aperture area	.8 cm ²
Transmission efficiency	60 %
Quantum efficiency	65 %
Signal integration time	1 sec
Pixel spread number	9
Dark current	500 e-/sec
Required signal to noise ratio	6

values of these variables. The main program then simply read the values from the files, eliminating the hideous task of reformatting after changing the values (as would have been necessary had the values been hard-wired into the program). The results are shown in Figures 4 & 5.

The studies show that the baseline values chosen were the values which should be used for further studies as determined by our programs capabilities. (The program which uses this data and actually displays the star scenes operates currently under a maximum number of ten thousand stars.) These values currently allow the "telescope" to detect 9119 stars.

However, this number is likely to change due to added features to the program. For example, the quantum efficiency had remained constant during these tests, but in actuality the quantum efficiency is dependent on the wavelength of observation. This feature was later added to the program but has not been used to redo these tests.

(Since the graphs shown retained a constant wavelength of .7 microns, the quantum efficiency would still remain constant but be at a slightly different value. The net effect would be a slight change in the number of stars detected while the overall shape of the graph remained the same.) A sensitivity study involving the waveband center wavelength was retested, though, using the variable quantum efficiency. Four different tables were used to determine the values of the quantum efficiency at the different wavelengths. The program allows the user to choose from these tables when determining the number of stars which can be detected. The result is shown in Figure 6.

After the completion of these sensitivity studies (more studies will be required at a future time), my next project was to add more features to the star detection program. The first feature enabled the program to determine doublet stars. Doublet stars are stars which are too close together to be differentiated by the telescope's sensor. The program defines doublet stars by their distance and magnitude as shown in Figure 7.

Figure 4.

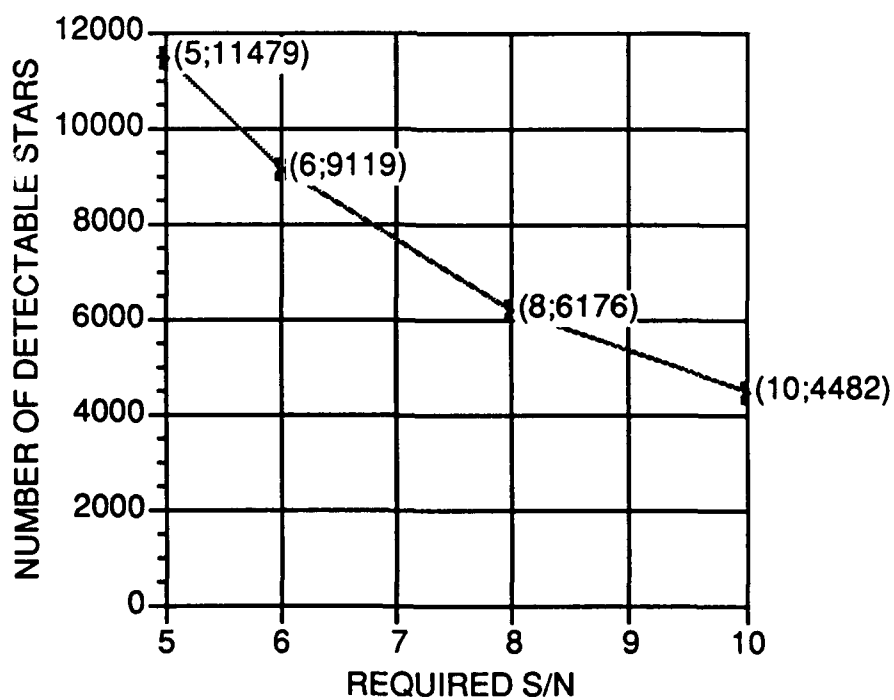
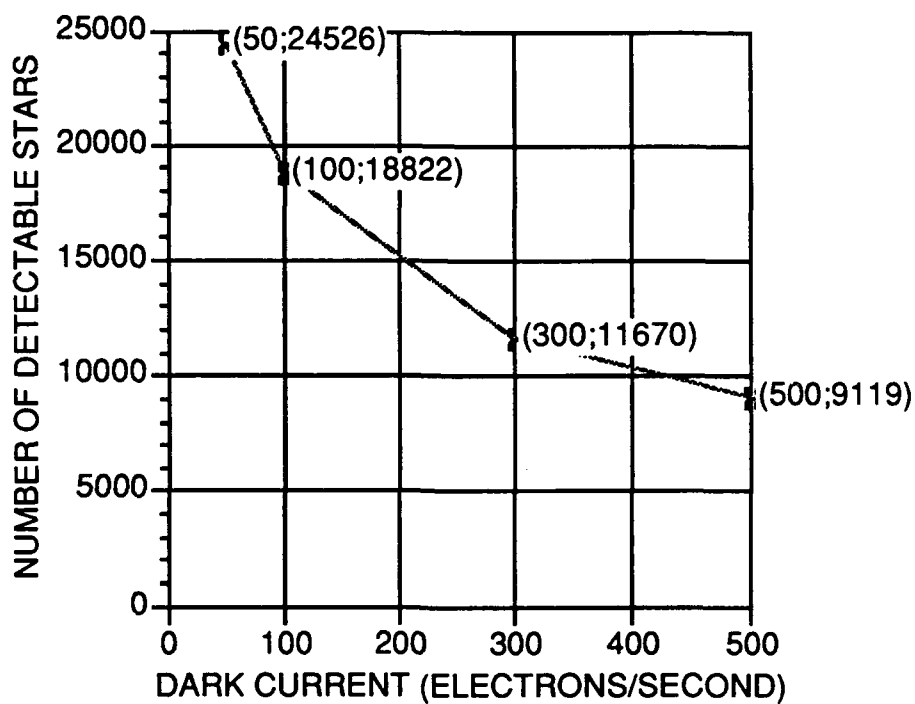


Figure 5.

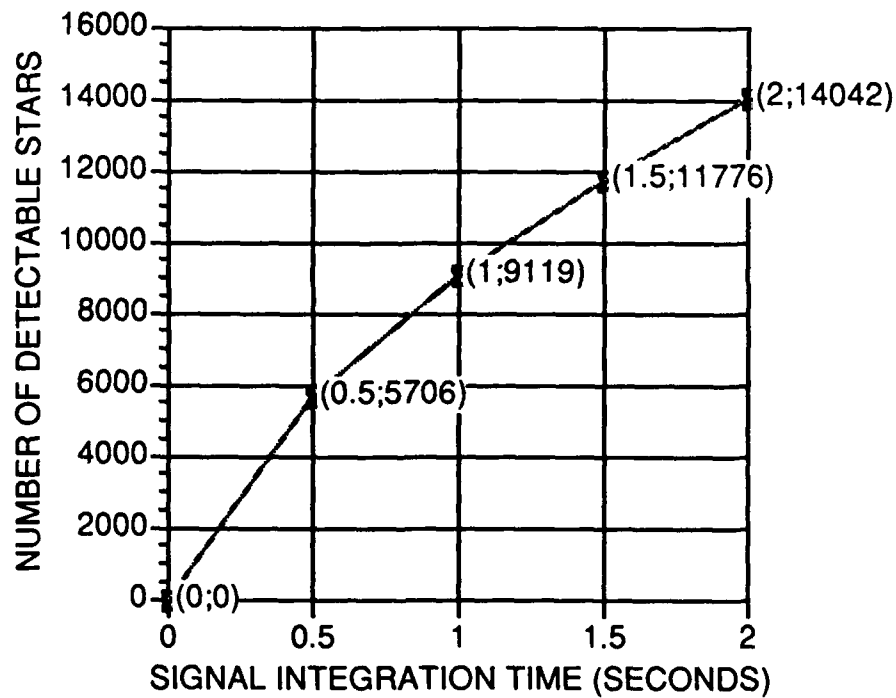
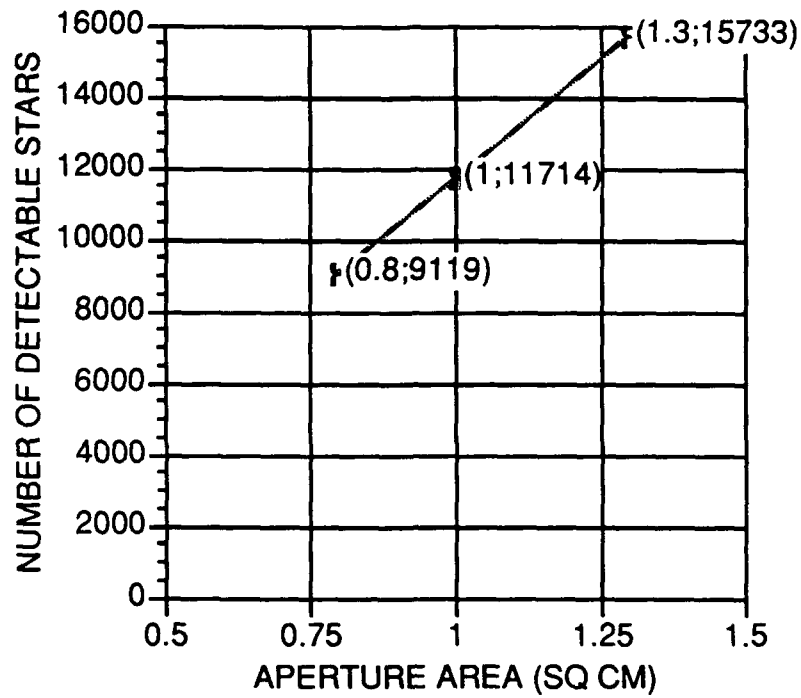


Figure 6
Number of Stars Detected versus Waveband Center Wavelength
Comparison of Quantum Efficiency Data Charts

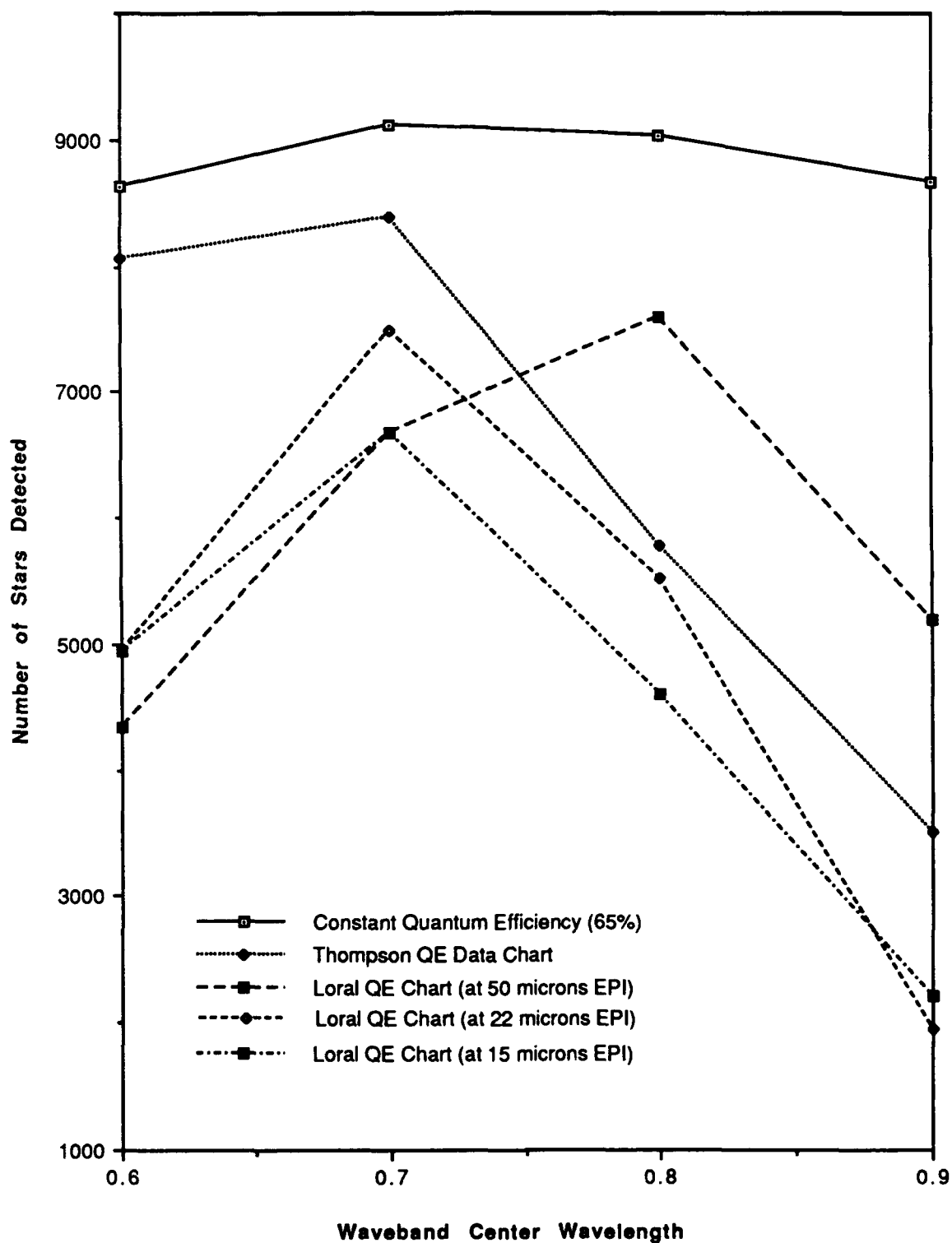


Figure 7

Determination of Doublet Stars

Stars are to be considered doublets if the following criteria are met:

- 1) Relative image separation on the FPA $< R$ arcsec
- 2) Stars too close in magnitude (< 6 magnitude difference)
- 3) $E \cdot (1 + 10^{.4(m_2 - m_1)}) < D$ where

E is the allowable centroiding error

$m_2 - m_1$ is the magnitude difference

D is the angular distance between the stars

and R is the resolution (IMV attitude accuracy)

The resolution was varied and resulted in a set of values which neatly fit a third-order polynomial curve (Fig. 8).

The second feature was a subroutine which calculates the probability of detection of any given star. This calculation is based upon the probability that a point on the signal's frequency curve will be higher than a point on the noise's frequency curve at a given time. This feature can determine whether a star can be depended on for detection, or whether it's signal will be drowned out by the noise.

IV. CONCLUSION

To conclude my project of star detection, I used a graphics program to create a plot of all the stars my program detected. The plot is shown on a celestial sphere at a view angle of 45 degrees. Another plot was made of stars which were detected by an application program which was simulating a telescope in orbit around the planet. This exercise provides a means of verification for that program. The two plots are displayed in Figure 9. The results of the plots, sensitivity studies, and other tests indicate that the

Figure 8
Number of Doublet Stars versus Resolution

Curve drawn in an approximation based on the equation:

$$y = 23.978 + 0.14413x + 5.7652e-4x^2 - 6.8243e-8x^3$$

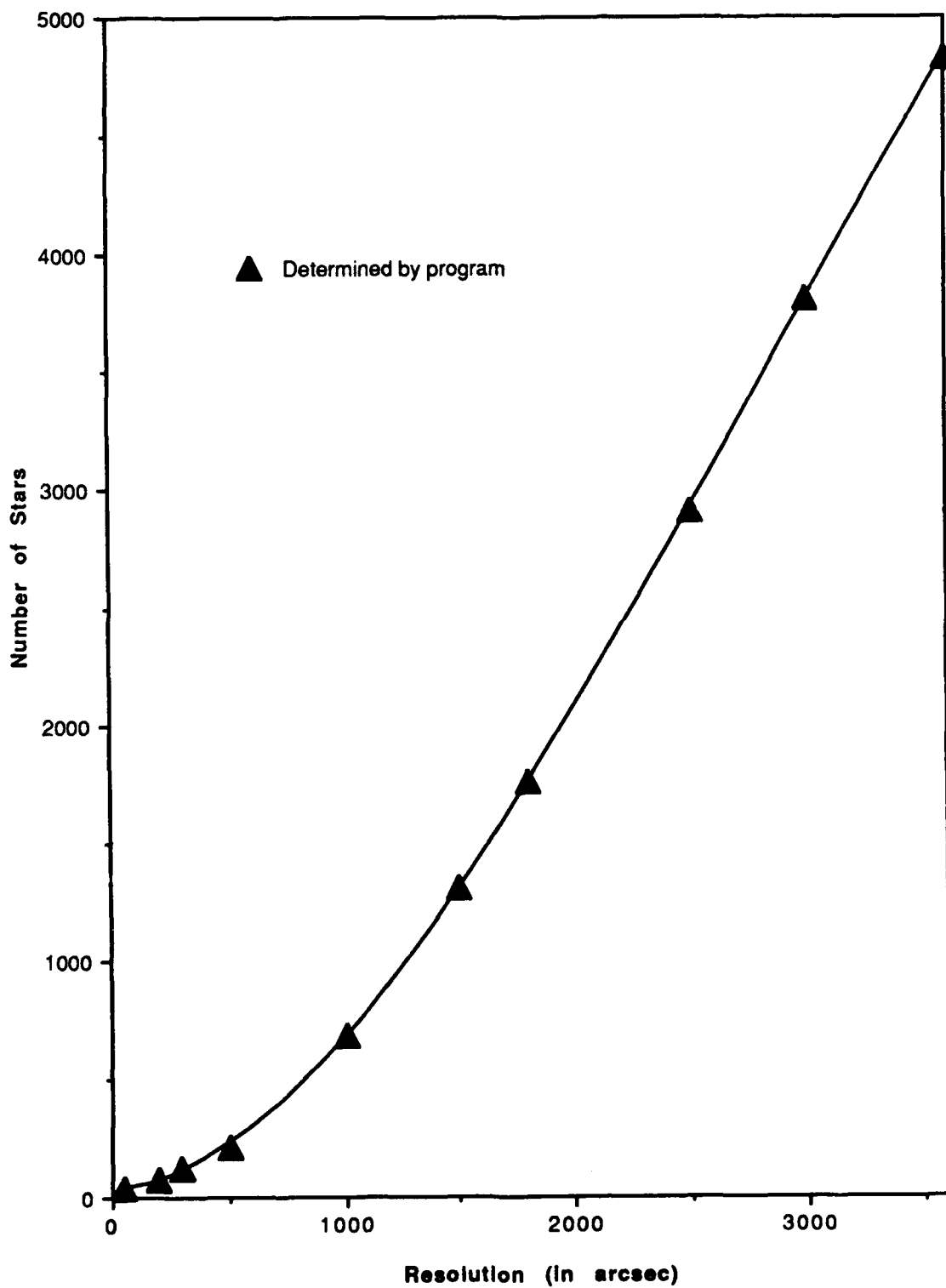
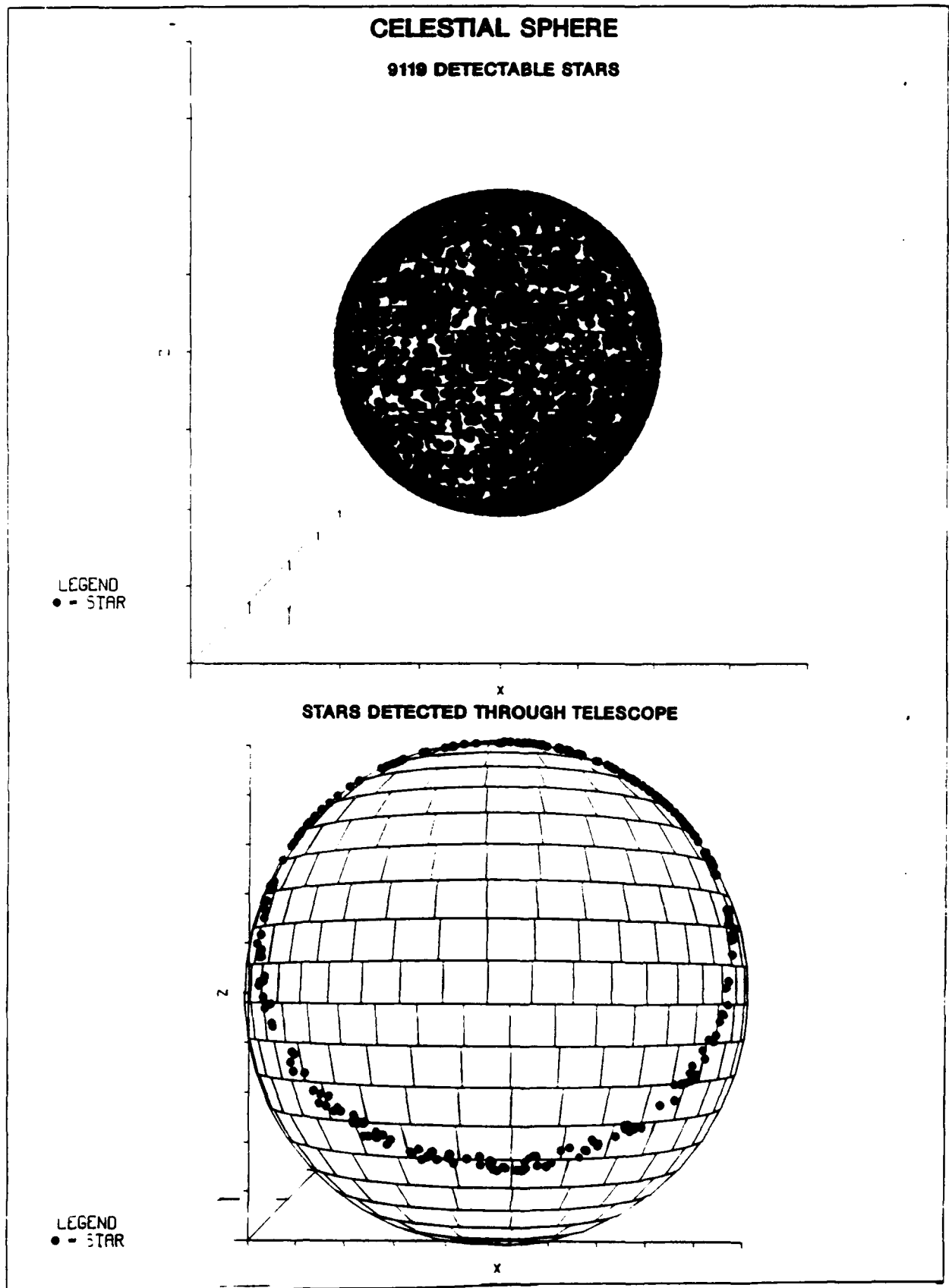


Figure 9.



programs are relaying accurate information. Additional subroutines will be implemented in the program to provide for enhanced flexibility and increased user-friendliness. The final product should allow ease of use and a multitude of possible tasks and perhaps combine the separate programs into one program. The programs as they currently stand are technically a complete project, however. The knowledge I have gained from them I will keep forever, and I know that I will use the knowledge in my future career in physics. I have thoroughly enjoyed this job and hope to come back next summer.

ACKNOWLEDGEMENTS:

I would like to thank Don Harrison for his hospitality and making me feel welcome in the apprenticeship program. I would also like to thank Randy Wells for allowing me to use his Macintosh and explaining to me the physics behind the programs I was writing. I would like to thank Mike Couvillon for also helping me with my computer problems. Lastly, I would like to thank Captain Tim Poth for making me feel at home in his office and mostly for being my mentor. Thanks to all of you and the rest of the MNSI staff, I have had the best summer of my life and am anxious to do this all again next year.

Appendix: Derivation of My Formula

This section is strictly mathematics, so I will try to explain each step I used to derive my equation. My rationale behind the formula was that if I were to convert any magnitude from the V-Band to the V-Band (yes, it's the same band, but the equations don't know that), then the resulting magnitude should be equal to magnitude I started with. Here are the steps:

Step 1: Start with the basic formulas which were used to convert a magnitude to another band. The only difference is that instead of SMAG, the left side of the equation is once again the VMAG and LAM is equal to .55 (the center wavelength for the visual magnitude).

$$\text{VMAG} = -2.5 \log \frac{H_{(\lambda)}}{H_0}$$

$$H_{(\lambda)} = H_{\text{peak}} \frac{Q^5}{21.199(e^Q - 1)}$$

$$Q = \frac{14387.9}{.55T}$$

$$H_{\text{peak}} = (H_{0\text{peak}}) 10^{-\text{VMAG}/2.5}$$

$$H_0 = 3.63078 * 10^{-12}$$

Step 2: Substitute for $\frac{H_{(\lambda)}}{H_0}$

$$\text{VMAG} = -2.5 \log \frac{H_{0\text{peak}} 10^{-\text{VMAG}/2.5} \frac{Q^5}{21.199(e^Q - 1)}}{3.63078 * 10^{-12}}$$

Step 3: Separate the $10^{-\text{VMAG}/2.5}$ and combine the denominator.

$$VMAG = -2.5 \log(10^{-VMAG/2.5} * \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}})$$

Step 4: Transform the product inside the log into a sum of two logs.

$$VMAG = -2.5 \log(10^{-VMAG/2.5}) +$$

$$-2.5 \log \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}}$$

Step 5: Simplify the left logarithm.

$$VMAG = VMAG - 2.5 \log \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}}$$

Step 6: Subtract VMAG from both sides.

$$0 = -2.5 \log \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}}$$

Step 7: Divide by -2.5.

$$0 = \log \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}}$$

Step 8: Take the antilog of both sides.

$$1 = \frac{H_{0peak} Q^5}{21.199(e^Q - 1) * 3.63078 * 10^{-12}}$$

Step 9: Isolate H_{0peak} .

$$H_{0peak} = \frac{21.199(e^Q - 1) * 3.63078 * 10^{-12}}{Q^5}$$

Step 10: Replace Q^5 with its value.

$$H_{0\text{peak}} = \frac{21.199 * 3.63078 * 10^{-12} (e^Q - 1)}{(14387.9 / .55T)^5}$$

Step 11: Combine constants and replace Q with its value.

$$H_{0\text{peak}} = 6.28263317702 * 10^{-33} T^5 (e^{14387.9 / .55T} - 1)$$

REFERENCES

Allen, C.W., "Astrophysical Quantities", third edition.

Holman, J. P. "Heat Transfer", fifth edition, 1981.

Warren, Wayne H. Jr. and Kang, Young Woon, "Skymap Catalog of 248576 stars", Version 3.3, August 1987.

Wyatt, Stanley P., "Principles of Astronomy", third edition, 1977.

PREPARATION AND CHARACTERIZATION

OF

3-PICRYLAMINO-1,2,4-TRIAZOLE

Kathryn Diane Deibler
Mr Stephen A. Aubert, Mentor
August 15, 1991

INTRODUCTION

Section I

3-picrylamino-1,2,4-triazole (PATO) is a heterocycle under investigation as a possible high energy insensitive explosive. PATO has been synthesized in small quantities and characterized (reference 10). As a continued study of the preparation of PATO, a 796 gram scale synthesis was conducted. The synthesis was verified using instrumental analyses. PATO's solubility was tested in organic solvents. To increase the particle size, recrystallization was investigated using precipitation, evaporation, and Soxhlet extraction methods. Density, heat of formation, and heat of combustion of PATO were determined. Calculations were conducted to estimate performance parameters including detonation velocity, detonation pressure, and Gurney energy, using Becker Kistiakowsky Wilson (BKW), Jones Wilkins Lee (JWL), Kamlet Finger, and Gurney methods.

ACKNOWLEDGEMENTS

Section II

I would like to thank Mr Stephen A. Aubert for designing a project that involved hands on chemical engineering. I appreciate all the technical assistance from TSgt Russell G. Huffman. I would like to thank Mr Don Harrison and Dr Norm Klausutis for managing the HSAP program. Again, I appreciate all the time Mr Stephen Aubert gave to helping me with my project.

BACKGROUND

Section III

Accidental initiation of 1.1 hazard class munitions presents a great threat to United States Air Force facilities. The High Explosive Research and Development facility at Eglin Air Force Base is developing high energy insensitive explosives to alleviate this threat. Insensitive high explosives offer advantages in operational readiness and sustainability through reduced quantity distance storage and handling restrictions (reference 1). Munitions filled with such explosives may be stored closer to airfields where they are readily available for use. Transportation of these munitions is also less hazardous. Insensitive munitions have a low probability of reaction when subjected to thermal, impact, friction, or electrostatic stimuli enhancing their survivability. However, the explosive will detonate with great power upon initiation of its designed explosive train. Most importantly they have low shock sensitivity and hence are non-mass detonating.

Explosives are useful militarily for applying disruptive energy to a target by converting chemical energy into kinetic energy. Chemical energy is applied by imparting kinetic energy into high velocity case fragments and expanding product gases. The combination of high energy output and insensitivity to shock initiation are highly desirable but difficult to obtain simultaneously with the present energetic materials available and conventional explosive formulation techniques.

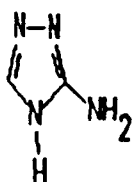
Heterocycles are organic ring compounds with atoms other than carbon making up the skeletal ring (reference 3). Azoles are five

membered heterocycles with one or more nitrogens. Heterocyclic materials are ideal for insensitive high explosives. Replacement of carbon with nitrogen provides the advantages of a higher density, favorable oxygen balance, and a positive heat of formation. High density improves a materials energy per unit volume, and hence its performance. Fuel, carbon and hydrogen, in an explosive reacts by combining with oxygen, producing carbon dioxide and water. A favorable oxygen balance, where all carbon and hydrogen is consumed with oxygen, results in more efficient reactant to product conversion and hence greater energy release. In the heterocycle, 3-picrylamino-1,2,4-triazole (PATO), nitrogens replace the carbons that would not have reacted due to an unfavorable carbon oxygen ratio. In addition aromatic nitro linkages decrease the compounds' sensitivity. High energy from the favorable oxygen balance, high density, favorable heat of formation, and insensitivity from the aromatic nitro linkage make heterocycles promising candidates for insensitive explosive applications.

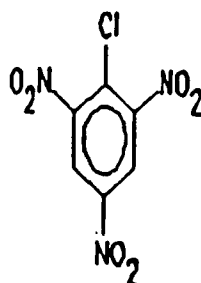
3-Picrylamino-1,2,4-Triazole (PATO) was first produced at the Los Alamos National Laboratory (LANL), and was characterized as a high density insensitive material (reference 2). Its energy was insufficient to meet nuclear applications, hence, only preliminary testing was completed. The triazole in PATO gives it the advantages of a heterocycle. PATO is synthesized by amination of picryl chloride (2,4,6-Trinitrochlorobenzene) with 3-amino-1,2,4-triazole (ATA) in dimethylformamide (DMF) at 100 degrees Celsius for a period of five hours (reference 2). The aryl halide, picryl chloride, and the amine, ATA, undergo a two step bimolecular displacement

nucleophilic aromatic substitution (reference 3).

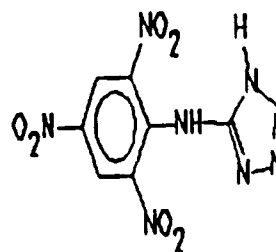
An aryl halide contains an aromatic ring with an attached halogen, an element in group VIIa on the Periodic Table. An aromatic compound is an organic ring with alternating double bonds. Orbital overlap of the alternating double bonds results in a resonance structure forming a delocalized pi cloud of electrons. Benzene carbon-carbon bond lengths are all equivalent and half the distance between single and double bond length. The heats of hydrogenation and combustion of benzene are lower than expected. These physical properties are a result of the delocalized resonance structure. Picryl chloride (2,4,6-trinitrochlorobenzene) is the aryl halide used in the synthesis of PATO. Picryl chloride's nitro groups (NO_2) located in the ortho and para positions of the benzene ring (2 and 5) are strongly electron withdrawing (reference 3).



ATA

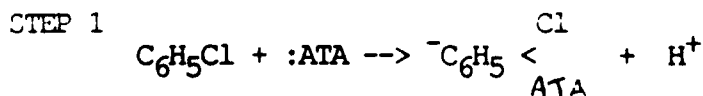


Picryl chloride

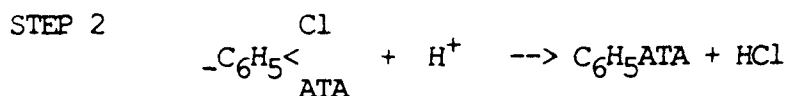


PATO

The first step of the two step mechanism is the attack of the nucleophile, ATA, on the aryl halide bond. The nucleophile has a lone pair of electrons located at the primary nitrogen. The first step forms a carbanion which is a negatively charged aromatic ring, halogen, and amine. This step is the slow, rate limiting and rate determining step.



Nitros attached to the aromatic ring increase the rate of formation of the carbanion by dispersion of its negative charge. Nitros, nitrides, acid, and aldehyde attachments are electron withdrawing and delocalize the negative charge further stabilizing the carbanion. The carbanion is a full negative ion formed by the amine attaching at the same bond as the halide. During the second step the halogen, chlorine, dissociates from the aromatic ring with a hydrogen forming hydrochloric acid and the aryl amine (reference 3).



The preparation of aniline from chlorobenzene is well known. $\text{C}_6\text{H}_5\text{Cl} + \text{NH}_3$ under 15 PSI at 1500°C yields $\text{C}_6\text{H}_5\text{NH}_2 + \text{HCl}$. The preparation of nitroaniline from nitrochlorobenzene is known to occur much more readily. $\text{C}_6\text{H}_4\text{NO}_2\text{Cl} + \text{NH}_3$ in DMF at 100°C yields $\text{C}_6\text{H}_4\text{NH}_2\text{NO}_2 + \text{HCl}$. PATO with two more nitro groups can be expected to react more readily.

The detonation of an explosive proceeds with the motion of the detonation wave through a reactive medium. The detonation wave consists of a shock front, reaction zone, and product expansion wave (Taylor wave). When a shock wave travels through an energetic material, the material is compressed by the shock front resulting in heating, exothermic chemical decomposition, and initiation of detonation. The chemical reaction occurs between the shock front and CJ plane, in a short but finite reaction zone. The expansion of the reaction products applies work to its surroundings in an irreversible process. The detonation performance of explosives is

measured by the resulting detonation velocity, pressure, critical diameter, and the Gurney expansion constant. The critical diameter is the minimum diameter which will support steady state detonation.

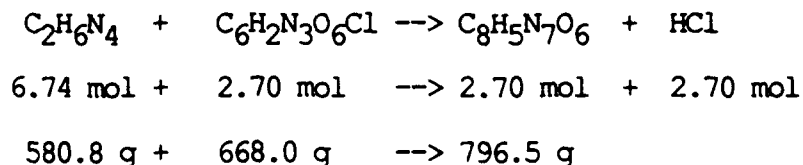
The results from the mass spectrum, NMR scans, elemental analysis, infrared spectrum, scanning electron micrograph, and x-ray spectrum have verified the synthesis of PATO. PATO is thermally stable with a 318.72 degrees Celsius decomposition exotherm and a critical temperature of 282°C. It does have an unfavorable small particle size which may be overcome through recrystallization. Preliminary tests show that PATO is insensitive to impact and high in performance, making it a promising candidate as an insensitive high explosive.

PROCEDURES

Section IV

PATO was synthesized by reacting 3-amino-1,2,4-triazole, ATA (580.8 g), and picryl chloride (668.0 g) in 6.5 L of N,N-dimethylformamide (DMF) in a 50 liter reaction vessel. The solution was heated at 100°C using a heating mantle and variac control for five hours and was continuously stirred by an air driven stirrer.

About 3.8 L of deionized water near 0°C was poured into the solution after the completion of the reaction to induce precipitation of the PATO. The precipitated solid was collected by vacuum filtration. The precipitate was dried in a vacuum oven for 24 hours. After grinding the material with a mortar and pestle, it was washed in around 9 L of deionized water near 0°C. The reaction equation and stoichiometry are as follows:



The solubility of PATO in dimethyl sulfoxide, cyclohexanone, and buytrolactone was tested by adding 0.1 g increments of PATO until the solution became saturated. This procedure was followed at room temperature and again after raising the temperature to 90°C. The solutions were heated indirectly by circulation and were continually stirred by a magnetic bead. After saturation was achieved at the higher temperature, the solution gradually cooled. The particles that crystallized upon cooling, were filtered.

The particle size of PATO had been found to be 1.26 microns. The volume of larger particles have less total surface area

resulting in a higher bulk density. The unfavorably smaller particle size consumes more volume with a greater total surface area than a larger particle size would. To solve this problem recrystallization by evaporation and by extraction using a Soxhlet extractor were attempted in dimethyl sulfoxide (DMSO). For the evaporation, 28.0 g of PATO were added to 100 mL of DMSO. The solution was continuously stirred by a magnetic stirring bead and heated using a variac heating mantel. The reaction was conducted under vacuum using a water aspirator. The solution was heated at 96°C for five hours. Recrystallization by Soxhlet extraction used 10 g of PATO powder, placed in a thimble in the extractor. DMSO (200 mL) and 60 g of PATO were added to the 500 mL round bottom flask. The extractor was attached to the top of the 500 mL flask. A condenser was placed at the top of the extractor and the system was under vacuum. The contents of the flask were heated between 70° and 85°C while being stirred with a magnetic stirring bar.

1.4 gram samples were pressed hydraulically in a half inch die at pressures ranging from 10,000 to 40,000 pounds per square inch. The resulting pellets' volumes were determined from the measured diameters and heights. The densities were determined using equation 1.

$$D = Wt / V \quad (\text{equation 1})$$

D = density
Wt= weight
V = volume

After receiving unfavorable pressing results of half inch diameter PATO pellets, a 95/5 weight percent coating of PATO and Kel-F-800 was conducted. Stirred by a air driven stirrer, 20 g of Kel-F-800 were added to 400 mL of ethyl acetate at 50°C in a

jacketed, one liter reactor. When the solution reached 70°C, 380 g of PATO were added along with 850 mL of ethyl acetate. Excess ethyl acetate was removed by evaporation under vacuum and trapped in a side flask.

Nuclear magnetic resonance spectrometry and thermal analysis with a differential scanning calorimeter were used for identification of the material. The density of PATO was determined using gas picnometry. Heat of formation was derived from the heat of combustion determined from bomb calorimetry.

Carbon-13 NMR spectra were obtained using a Bruker AC-300, 300 MHz Fourier Transform Superconducting NMR spectrometer. Two 3 mg samples of PATO, one dissolved in a milliliter of deuterio DMF and another sample dissolved in deuterio DMSO, were each placed in 5 mm glass NMR tubes. The resonance frequency (ν) is determined by the fundamental NMR equation $\nu = \gamma / 2\pi H^0$, where γ is the gyromagnetic ratio and H^0 is the main field strength. Nuclear magnetic resonance spectroscopy measures the applied field strength plotted against the absorption signal. In NMR a superconducting magnet produces a homogeneous magnetic field of approximately 7.4 Tesla between its poles. The sample is spun about its main field (Z) axis by a stream of air to average out any existing homogeneities in the xy plane. The number of signals on the spectrum tells how many "kinds" of like atoms there are in the molecule. The position of the signals indicates the electronic environment of each atom. The splitting of a signal into several peaks shows the environment of a carbon with respect to other, nearby carbons (reference 3).

PATO samples (.5 mg) were thermally analyzed on a Perkin-Elmer

System-4 Differential Scanning Calorimeter. Heat was applied at a rate of 10.00 degrees per minute. The scan produces a trace of the endothermic and exothermic events which take place as the temperature of the sample rises in an inert N₂ environment (reference 6). The samples were crimped in either vented or air tight aluminum sample pans.

Density was determined by gas pycnometry using a Quantachrome model PP-4 penta-pycnometer. Five samples in small cells, were run with a one minute purge time.

The heat of combustion was determined by bomb calorimetry. Five 1.0 g samples of PATO pressed to a density of 1.72 g/cc were burned under 30 atmospheres of oxygen using the Parr Bomb Calorimeter. The heat of formation was derived using Hess's Law, where the heat of reaction is equal to the difference between the heat of formation of products and reactants. The heat of combustion for Tritonal and TNT were also determined and used as standards of comparison. The equations for the calculation of the heat of combustion and the heat of formation are shown in the following table.

3-210714-1,2,4-714718/700

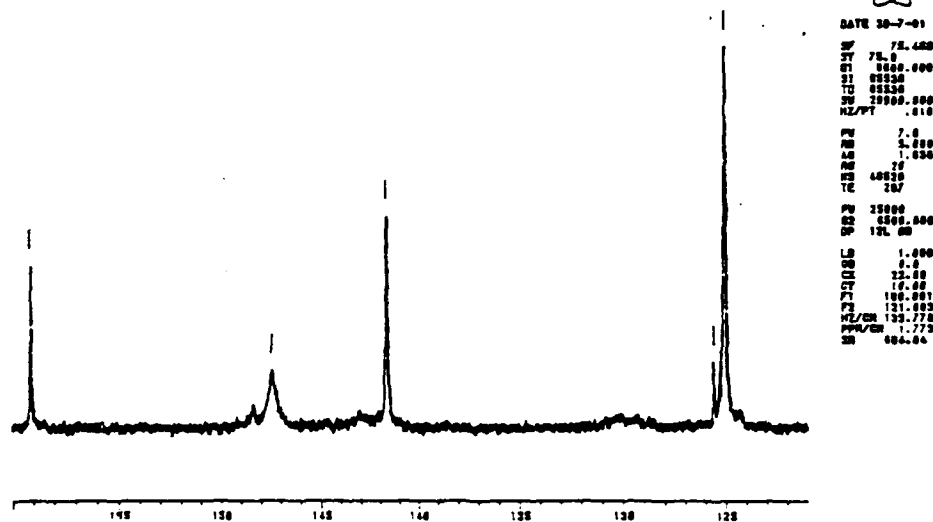
[illegible]

FIGURE 1 NMR SCAN

The onset of the exotherm on the differential scanning calorimeter (DSC, figure 2) was found at 319.3°C. The decomposition temperatures, onset temperatures and area under the curve obtained from the DSC are as follows:

DSC RESULTS

Sample	Decomposition	Onset	Area
vented pan	322.1°	319.33°	148.99°
unvented pan	326.8°	324.95°	102.86°

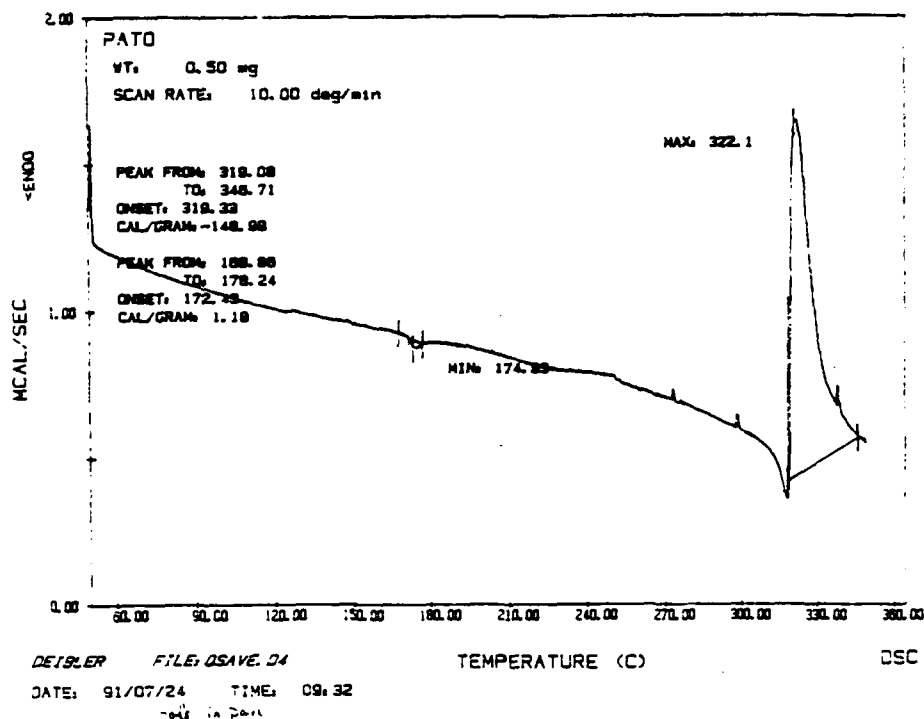


FIGURE 2 DSC SCAN

The average density of PATO was found to be 1.935 ± 0.001 g/cc. The density results for each run appear in the table below.

DENSITY

Weight = 2.420 g

Run	Volume (cc)	Density (g/cc)
1	1.251	1.935
2	1.252	1.932
3	1.249	1.937
4	1.250	1.936
5	1.251	1.935
Average		1.935 ± 0.001 (1.94)

The experimental heat of combustion and derived heat of formation are listed in the table below. The heat of combustion for PATO had a precision of 0.03 percent while the heat of formation was precise to 1.5 percent. The value for TNT was within .3 percent of that found in literature while the Tritonal value was within 3.2 percent of that in the literature.

HEAT OF COMBUSTION AND HEAT OF FORMATION CALCULATIONS

Heat of Combustion $H_{\text{comb}} = W(dT) - e_1 - e_2 - e_3 / \text{wt sample}$
 W = Energy Equivalence of calorimeter
 dT = temperature rise
 e_1 = Correction for heat from HNO_3 formed
 e_2 = Correction conversion from constant volume to constant pressure
 e_3 = Correction for heat from fuze wire

Heat of Formation $H_{\text{RXN}} = H_{\text{form products}} - \text{reactants}$

$$H_{\text{form reactant}} = H_{\text{form products}} - H_{\text{comb}}$$

Using Tiger program and Kamlet simplified method, detonation velocity (D_v), and detonation pressure (P_{CJ}), were calculated based on the experimental density and heat of formation. The 2E was determined by Gurney calculator using Hardesty Kennedy and Kamlet Finger methods.

RESULTS

Section V

When ATA, picryl chloride, and DMF were added to the reaction vessel, the liquid gradually turned maroon in hue. Yellow crystals formed and fell to the bottom as the cold water was added. The wet weight after the first washing was 828.5 g. After washing the 715 g of PATO particles again, the yield was 603.4 g, 34.4%.

In testing solubility, 1.2 g of PATO added to 100 mL of DMSO at 30°C clouded the solution. The temperature was raised, then kept constant at 88°C. The solution became cloudy when 28.26 g of PATO were in solution, and the solution became clear when the temperature was raised to 92°C. The solution was cooled down at an average rate of .72 degrees per minute and filtrated 24.8 g of PATO.

Less than .1 g of PATO was soluble in 100 mL of cyclohexanone at 26°C. 1.5 g of PATO saturated 110 mL of cyclohexanone at 98°. The maroon-yellow colored solution cooled at .4823 degrees per minute to 21°C and was left for two days. The PATO was still in solution, while about two thirds of the cyclohexanone had evaporated, leaving a slurry of undissolved PATO and solvent.

.39 g of PATO saturated 100 mL of butyrolactone at 26°C. 3.35 g of PATO saturated the solution at 90°C. The solution cooled at .75 degrees per minute. Particles began to form out of solution around 57°C. The solubility results are summarized in the following table.

Solvent	SOLUBILITY	
	Temperature	Solubility g/L +/- 1 g
Butyrolactone	26°C	3.9
	90°C	33.5
DMSO	30°C	12.14
	92°C	28.26
Cyclohexanone	26°C	<1.0
	98°C	13.64

The solution in the evaporation crystallization of 28.0 g of PATO in DMSO became saturated around 60°C and boiled at 96°C. Minimal crystals formed directly from the evaporation. Crystals formed when the solution was cooled down. These particles that formed were apparently not significantly larger than the original particles. No crystals formed from the Soxhlet extraction.

RECRYSTALLIZATION

Method	Solvent		
	Cyclohexanone	Butyrolactone	DMSO
	Particle Size		
Evaporation	< 2 u	< 2 u	< 2 u
Extraction	-	-	< 2 u

The Carbon-13 NMR Spectrum of PATO in DMF (figure 1) contained peaks at field shifts of 141.85, 140.20, 125.81, and 125.08 ppm. These peaks correspond to the absorption of the picryl carbons. Peaks at field shifts of 147.42 and 158.12 ppm correspond to carbons in the triazole ring. The spectrum of PATO in DMSO contained peaks at field shifts of 124.98, 125.53, 140.5, and 142.97 ppm which correspond to the picryl carbons. Peaks at field shifts of 153.40 and 155.90 ppm correspond to the carbons in the triazole ring. Extra peaks in the spectrum of PATO in DMSO indicated an impurity of DMF in the PATO, which led to a thorough washing of the material.

HEAT OF COMBUSTION AND FORMATION DATA

Explosive	Heat of Combustion (cal/g)		Heat of Formation (kcal/mol)
	Measured	Literature	Measured
TNT	3630.3	3620	-
Tritinal	4338.2	4480	-
PATO Run 2	3235.5	-	+18.3
PATO Run 3	3233.0	-	+17.6
PATO Run 5	3232.3	-	+17.4
PATO Avg	3233.6 <u>±</u> 1.0	-	+17.8 <u>±</u> 0.3

BKW codes with TNT parameters were used for performance calculations because PATO is fuel rich like TNT. The detonation velocity (D_v) calculated to be 7.744 mm/usec and the detonation pressure (P_{cj}) calculated to be 285.7 kbar. Using Kamlet simplified method the detonation velocity calculated to be 7.73 mm/usec and the detonation pressure was 272.9 kbar. The two methods of calculating detonation velocity and detonation pressure agree 99.85% and 95.52% respectively. The Gurney Energy calculated to be 2.494 mm/usec compared with 2.40-2.44 mm/usec for TNT using Kamlet Finger method. The wall velocity at 19 mm calculated to be 1.557 mm/usec and the specific energy at 19 mm calculated to be 1.246 MJ/kg.

PERFORMANCE CALCULATION

	PATO	TATB *
Detonation Velocity (mm/usec)		7.62 *
BKW	7.74	
Kamlet Simplified	7.73	
Detonation Pressure (kbar)		259 *
BKW	285.7	
Kamlet Simplified	272.9	
Gurney Energy (mm/usec)	2.49	2.38 *
Density (g/cc)	1.94	1.85

* Measured values

CONCLUSIONS

Section VI

The large scale synthesis of PATO was successfully achieved with a high yield of 84.4%. The results of the differential scanning calorimeter spectrum and the Carbon-13 NMR scans verify the compound synthesized is PATO. Although PATO is more soluble in dimethyl sulfoxide (DMSO) than in either butyrolactone or cyclohexanone, recrystallization to a larger particle size in DMSO by conventional methods was unsuccessful. The detonation velocity was calculated to be 7.7 mm/usec and the detonation pressure was found to be 273 kbar, almost equivalent to that of TATB. Since PATO synthesized particles are too small and recrystallization to a larger particle size was unsuccessful, PATO may be used for an insensitive booster similar to PBX-9502 a combination of 95/5 TATB/Kel-F-800.

MISCELLANEOUS

Section VII

This summer through the HSAP I learned the importance of patience and precision in conducting scientific experimentation. I have a more complete understanding of a government experiment program since I have worked for two summers on the same project. I gained experience with actual chemical engineering procedures which will be useful as I begin studying chemical engineering.

REFERENCES

Section VIII

1. Aubert, Stephen A. Interim Qualification of AFX-1100, AFATL-TR-89-48, Air Force Armament Laboratory, Eglin AFB, Florida, September 1989.
2. Cohurn, Michael D. 3-Picrylamino-1,2,4-Triazole and its Preparation. U.S. Patent 3 483 211, 1969.
3. Morrison, Robert T. and Robert N. Boyd. Organic Chemistry, Third Edition. Boston: Allyn and Bacon, Inc., 1973.
4. Silverstein, Robert M., G. Clayton Bassler and Terence C. Morrill. Spectrometric Identification of Organic Compounds, Forth Edition. New York: John Wiley and Son, 1981.
5. Ring, Terry A. "Making Powders." Chemtech, January 1988.
6. Perkin-Elmer. Instructions to Perkin-Elmer Model 3600 Data Station PETOS CS/12. Norwalk, Connecticut: Perkin-Elmer, 1983.
7. Gibbs, Terry R. and Alphonse Poplatoetal. "LASL Explosives Property Data." Berkeley, California: University of California Press, 1980.
8. Dobratz, B. M. "LLNL Explosives Handbook Properties of Chemical Explosives and Explosive Simulants." Livermore, California: University of California, March 16, 1981.
9. Kramer, Michael P., William P. Norris, and David J. Vanderah. "CL-14, A High-Performance Insensitive Explosive." China Lake, California: Office of Naval Research, June 1989.
10. Deibler, Kathryn D. "Synthesis and Characterization of 3-picrylamino-1,2,4-triazole." Eglin AFB, Florida: AFATL, August 15, 1990.

High School Apprenticeship Program (HSAP)
Research Paper

Mercury-Indium-Silver Alloys for Joining Copper
and Aluminum Striplines
by Lesha Denega

August 1991

This summer, I worked as a high school apprentice under the direction of Dr. Duane Finello at the Fuzes Branch of Wright Laboratories' Armament Directorate at Eglin Air Force Base. As a high school apprentice, I had the opportunity to learn valuable, hands-on information in the engineering and science fields. As the weeks progressed, I was able to put this information to valuable use. The project to which I directed the bulk of my time dealt with metallurgy and alloys. The goal of this project was to recreate a paint-like solder paste called alloy 232. I was to use X-Ray fluorescence (XRF) spectroscopy^{1,2} to perform a complete chemical analysis of this unusual solder using quantitative comparisons with a series of standards of known composition that I would prepare.

An understanding of the nature of 232 was necessary to recreate it. In 1985, Victor King Labs began manufacturing 232, and several years later they went out of business, taking 232 with them. A patent was never filed, and 232 disappeared from the technical market. A solder that is liquid at room temperature (not requiring as great a volume per joint as standard lead-tin solder) would be very beneficial at the Fuzes laboratory. Much of the work done at this lab involves lengths of ribbon-like copper conductors called striplines. As such, there

is a potential for practical weapon application of 232, but its composition was unknown. Earlier in the year a cursory XRF analysis gave an indication that silver (Ag) and indium (In) were present in this alloy (Fig. 1,2). The presence of mercury (Hg) (Fig. 3) was obvious due to the lustrous appearance of the alloy, which ruled out the possibility of gallium being the primary alloy constituent. Knowing this, I was to prepare numerous samples of known composition in an attempt to obtain several that closely resembled the physical properties of 232. I was to select homogeneous samples as standards and determine which ones most closely matched alloy 232 quantitatively in order to establish its composition. Unfortunately, severe technical difficulties associated with maintenance of the XRF spectrometer precluded its use for virtually the entire summer.

Knowing the elements in this unique solder, I gathered all relevant binary alloy data³ (Fig. 4,5,6) and familiarized myself with some of the physical properties of alloys in general. In the process I became aware of some underlying principles one must understand in order to make practical use of the published binary and ternary phase diagrams. Important factors which tend to influence the solubility of one element into another are known as the Hume-Rothery⁴ rules. There are three factors that influence this solubility:

"Size Factor- "A necessary condition for the formation of a solid solution when two metals are alloyed is that their atomic sizes or effective radii be within 15% of one another...but if

the size difference is greater than 15%, solubility will be very limited.

"A large size difference between solute and solvent means that there is a large elastic strain set up around each solute atom in the solid solution. With increasing solute content the strain energy in the solid increases making the solution unstable.

"Valence Factor - As the valence of the solute and the solvent become more unlike, solid solubility becomes more restricted. The valence difference of the solute and the solvent determines the electron atom ratio, e/a , of the alloy, i.e. the number of valence electrons present per atom. For example, $e/a = 1.5$ for an alloy of 50 at. % Cu (valence 1) with 50 at. % Zn (valence 2). Alloy structure can generally tolerate only a certain change before they become unstable or of too high energy and transform to another lower-energy structure. This is illustrated by alloys of copper with metallics of higher valence; as the valence of the solute increases, its maximum solid solubility decreases.

"Electronegativity- Elements such as F and Cl are strongly electronegative - meaning that they have a strong affinity for electrons when chemical bonds are formed. In metals such as Na and Mg, electronegativity is very low, while elements such as Pb and Sn near the center of periodic table have intermediate electronegativities. When the electronegativity difference between two elements is large, the elements tend to form

compounds of definite composition rather than solutions. Thus a large electronegativity difference between two elements means that the solubility of one in the other will be limited. Magnesium (Mg) and tin (Sn), for example, differ appreciably in electronegativity and form the compound Mg₃Sn. The solubility of Sn in Mg is only 3.4 at. %, even though the size factor is favorable."

"The description of alloy systems in terms of the above three factors is, of course, an oversimplification of a highly complex problem, but it is frequently useful and often allows estimates of phase relations in new alloy systems for which diagrams have not been worked out."

Knowing this I set out to create a rough ternary phase diagram^{3,5} for the unknown alloy (Fig. 7), which would cover a wide range of temperatures and all compositions. This would not be very detailed but it would be helpful in plotting some estimated composition ranges at room temperature. I also looked at the three elements with regard to the aforementioned Hume-Rothery rules. By these rules, a correlation between the Ag-Hg-In system and the Ag-Hg-Sn system can be noted:

(1) Working radius - The covalent radii of Sn and In are very close; .144nm for In and .141nm for Sn. Their atomic radii are within the required 15% for optimum solubility with In at .200nm and Sn at .172nm. Sn is actually closer to Ag and Hg with regard to atomic radii. Thus, In may be a little less

soluble than Sn with regard to Hg and Ag.

(2) Valence- Ag is univalent , Hg is divalent, and In is trivalent. Sn has a valence of 4, but this may balance with the radius and electronegativity factors, thus giving In and Sn nearly the same behavior, respectively, in a ternary alloy with Hg and Ag. The higher the valence difference, the less soluble the elements are together.

(3) Electronegativity- Sn has an electronegativity of 1.96 (Pauling's), close to In (1.78) but even closer to Ag (1.93) and Hg (2.00). This third factor compensates for some of the valence difference.

Using these similarities, I was able to base some of my ternary Ag-Hg-In data on the established ternary for the dental amalgam, Ag-Hg-Sn. Doing so, I could create ternaries, eutectics and an isotherm with a greater degree of accuracy,

It was decided that it would be most appropriate to proceed using qualitative analysis. I began creating samples of known composition using 99.99% pure - 30 mesh Ag powder (Aldrich Chemical Company), pure In foil, and electronic grade mercury (Aldrich Chemical Company). The compositions of these samples is expressed in Table 1. Some samples were made using In powder that we later discovered was impure; these were discarded for data purposes.

It must be noted that all of the samples were intermetallics, and not completely homogeneous. Alloy 232 itself

is not truly homogeneous, as there is a microscopic precipitate interspersed throughout the liquid. Some mixtures were less homogeneous than others; the ideal of course, is similar to 232, and is very close to being completely homogeneous.

According to the Ag-Hg-In phase diagram, you cannot add more than roughly 2% Ag to Hg at room temperature and have all of the Ag be absorbed into the liquid as a single phase. It must be noted that aggregate alloys formed from a higher temperature result in a finer dispersion of the alpha or beta precipitates in the two phase liquid. Higher amounts of In reduce the room temperature solubility of Ag in the single phase Hg + In liquid, and I discovered that 1% Ag or less was the highest amount of Ag that could be added to alloys containing amounts of In required to reduce the Hg surface tension. Greater percentages of Ag cause precipitates of unmanageable size to appear in the mixture.

These higher amounts of In are necessary in order for the alloy to have the property of wettability. Wettability is necessary because it allows the alloy to be spread on the surface of the joint with greater ease. Percentages of indium of at least 50% will produce the most desirable qualities (including reduced surface tension and increased viscosity) which enhance overall wettability.

Since much of my analysis was qualitative, an equation was developed to compare the conductivities of the samples against each other. This equation (expressed in Figure 8) is based on the rule of mixtures, and it is dependant on the volumetric

ratios of each element to the other in the alloy (Table 2). The results are not completely accurate for practical purposes, but they served as a precomparison of relative conductivities in the sample I created. The results are not ideal, yet serve to project relative conductivities.

The next step in this project was to join copper (Cu) stripline material to itself using several candidate alloys, with a 60% Sn/40% Pb solder joint as a comparative standard. After many failures, it was discovered that the key to making well-bonded joints was the pretreatment of the Cu surface. Thus, the energy barrier (activation energy) for diffusion of Hg into Cu is relatively high if the Cu surface is not clean with exposed crystalline defects. In order to expose the grain boundaries, etching the surface was necessary. First I prepared a solution of 200mL reagent water and 187g ammonium persulphate crystals. After heating and stirring, the crystals dissolved, and .3oz sulfuric acid was added. Each joint interface was dipped into this solution for 30 seconds, washed in ethanol, and then quickly dried under a heat gun. After proper surface preparation, the joint hardened readily since rapid transport of Hg into the Cu would take place. A description of this type of rapid diffusion phenomenon can be found in Reference 6.

The estimated strength of Cu/Cu alloy 232 joints is 80-100lbs/inch square. The joints I would make would only be able to support a force of roughly 1 pound, but they would be exceptionally thin in comparison to solder

After the success of making copper stripline joints with the alloy samples, joints were created using Al to Al and Cu to Al. Sn/Pb solder cannot join Al to Al or Cu to Al without nickel plating or an intermediate process called zincating. After creating these new joints, their conductivities and the copper joint conductivities could be prepared for later conductivity measurements. It was discovered that the Hewlett Packard LCR Meter was not effective for our purposes, and data gathered from those measurements was pertinent only in its relation to the other samples, not real values. Later I would use joint width information (Table 3) to compute these conductivities after qualitative measurements.

Until the XRF can be used to determine the quantities of the known elements in 232, the true composition will not be known. Regardless, two samples other than 232 exhibited qualities that would be very useful in practical weaponry systems. The bonding of Cu to Al is especially beneficial as it will allow an engineer to use aluminum components in conjunction with copper components in weapons configurations. Since I am externing with Dr. Finello, I plan to run quantitative analysis when the XRF is functional.

While this project was riddled with diverting setbacks, it provided me with more knowledge and experience than a year's worth of high school chemistry. I now have a better understanding of the relationship of metal in alloy systems, and of the behavior of liquid alloys with relation to Cu and Al surfaces. I would first like to thank my mentor Dr. Finello for

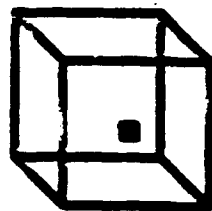
teaching me as much as he confused me. I would also like to thank Christine Riendeau, Charles Miller, Mr. Crews, Mr. Harrison, Dr Klausutis, all the folks at MNMF, and at MNGA Advanced Guidance. And last, but not least, I thank RDL and the USAF.

References

1. R. Jenkins "X-Ray Fluorescence Analysis" Analytical Chemistry, Vol. 56, No. 9, August 1984. p. 1099A-1105A.
2. M. Dasgupta, B.K. Sharma, B.L. Ahuja, and F.M. Mohammad "Some Experiments on X-Ray Fluorescence for the Student Laboratory" Am. J. Phys Vol 56 (3), March 1988. p 245-251.
3. Metals Handbook (8th Ed.), Metallography, Structures, and Phase Diagrams (Vol. 8), 1973 "Phase Diagrams for Binary Alloy Systems The American Society for Metals, Metals Park OH, pp. 252-434.

4. R.M. Brick, R.B. Gordon, and A. Phillips, Structure and Properties of Alloys (1965). McGraw-Hill Publ. Co., New York, pp 141-144.
5. E.M. Levin, H.F. McMurdie, and F.P. Hall, Phase Diagrams for Ceramists (1956). The American Ceramic Society, Columbus OH, pp. 14-25.
6. C.A. Keyser, Materials Science and Engineering (1980). Charles Merrill Publ. Co., Columbus OH, pp. 191-198.

Silver (Ag)



cubic, face centered

<i>atomic number</i>	47
<i>atomic weight</i>	107.868
<i>melting point</i>	961 C
<i>boiling point</i>	2163 C
<i>electronegativity</i>	1.93 (Pauling's)
<i>electrical conductivity</i>	0.630 [$10^6 \Omega^{-1} \text{ cm}^{-1}$]
<i>density at 300 K</i>	10.5 g/cm ³
<i>atomic radius</i>	0.175 nm
<i>covalent radius</i>	0.134 nm
<i>oxidation state</i>	1
<i>electron configuration</i>	[Kr] 4d ¹⁰ 5s ¹
<i>valence</i>	1

Indium (In)

atomic number 49

atomic weight 114.82

melting point 156.76 °C

boiling point 2073 °C

electronegativity 1.78 (Pauling's)

electrical conductivity $0.116 \times 10^6 \Omega^{-1} \text{ cm}^{-1}$

density at 300 K 7.31 g/cm^3

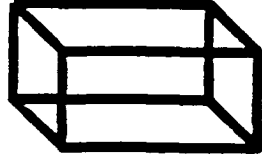
atomic radius 0.2 nm

covalent radius 0.144 nm

oxidation state 3

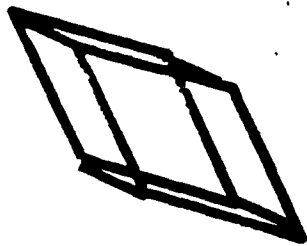
electron configuration [Kr] $4d^{10}5s^2 p^1$

valence 3



Tetragonal

Mercury (Hg)



rhombohedral

atomic number 80
atomic weight 200.59
melting point -38.72 C
boiling point 357 C
electronegativity 2.00 (Pauling's)
electrical conductivity $0.0104 [10^6 \Omega^{-1} \text{ cm}^{-1}]$
density at 300 K 13.53 g/cm^3
atomic radius 0.176 nm
covalent radius 0.149 nm
oxidation states (bold most stable) **2, 1**
electron configuration **[Xe] $4f^{14} 5d^{10} 6s^1$**
valence 2

Mercury-Indium Phase Diagram

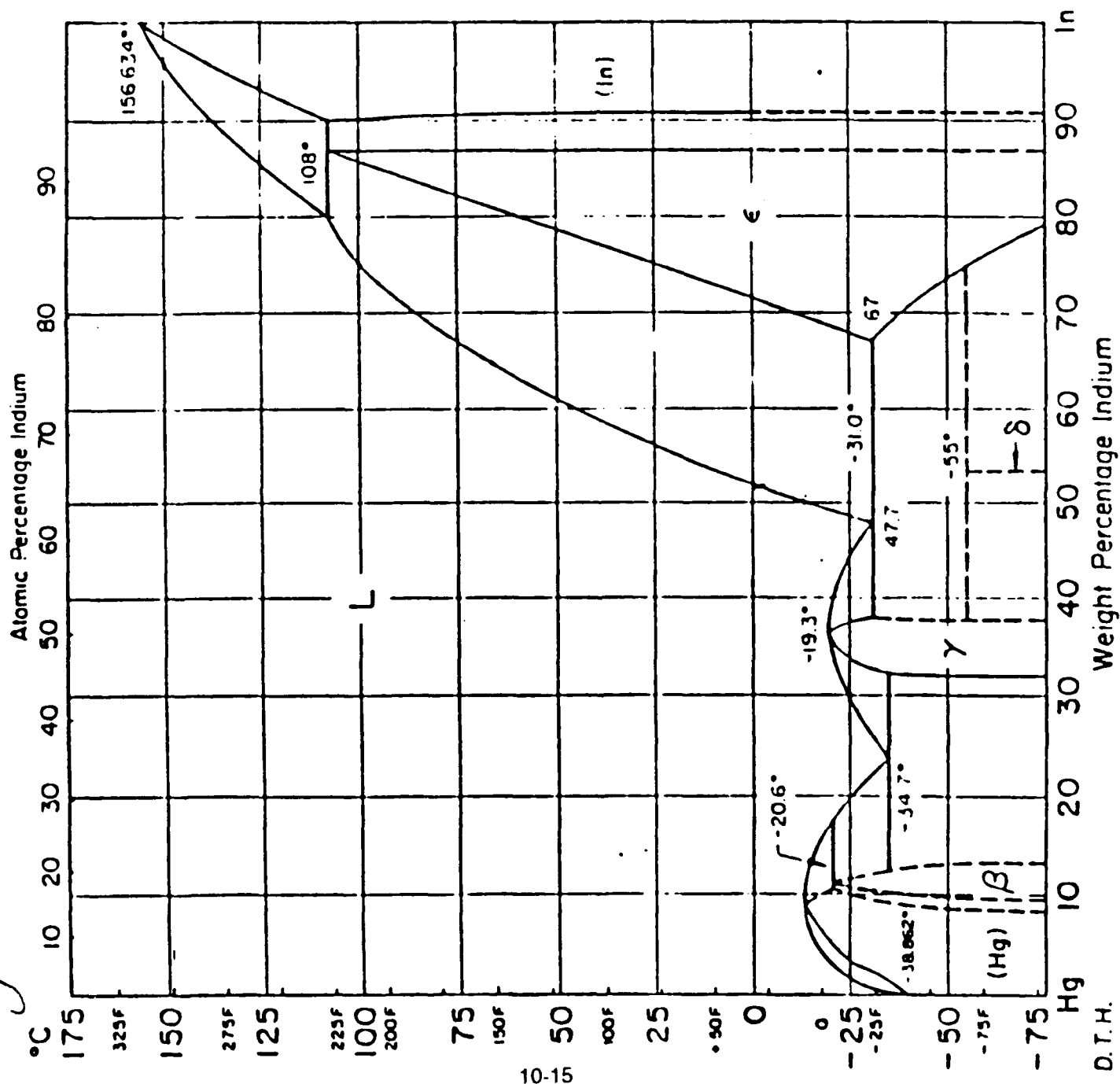


Fig. 2

Silver - Mercury Phase Diagram

Ag-Hg Silver-Mercury

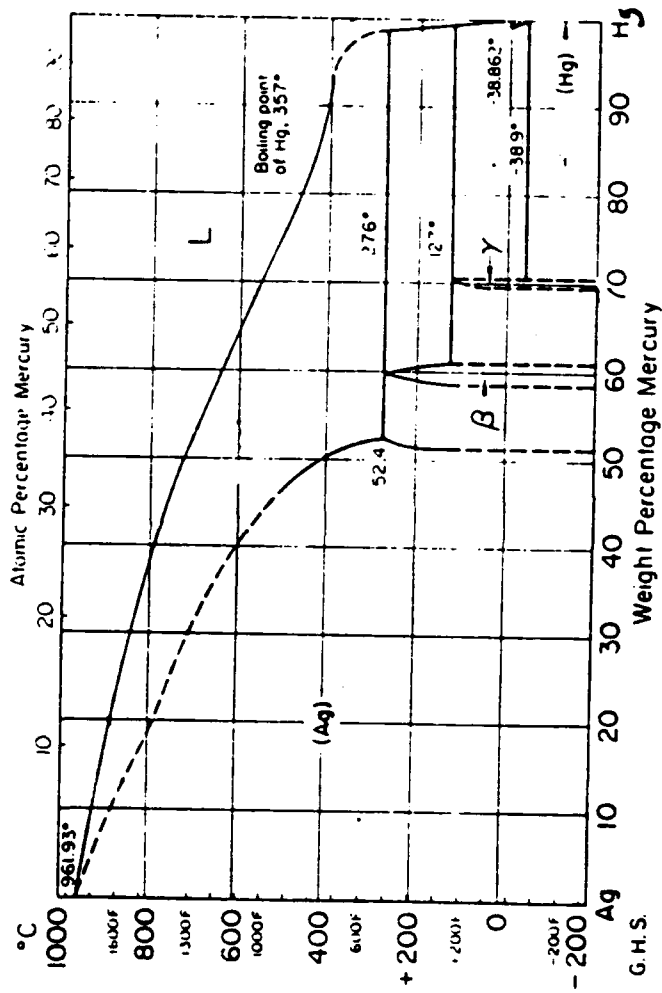
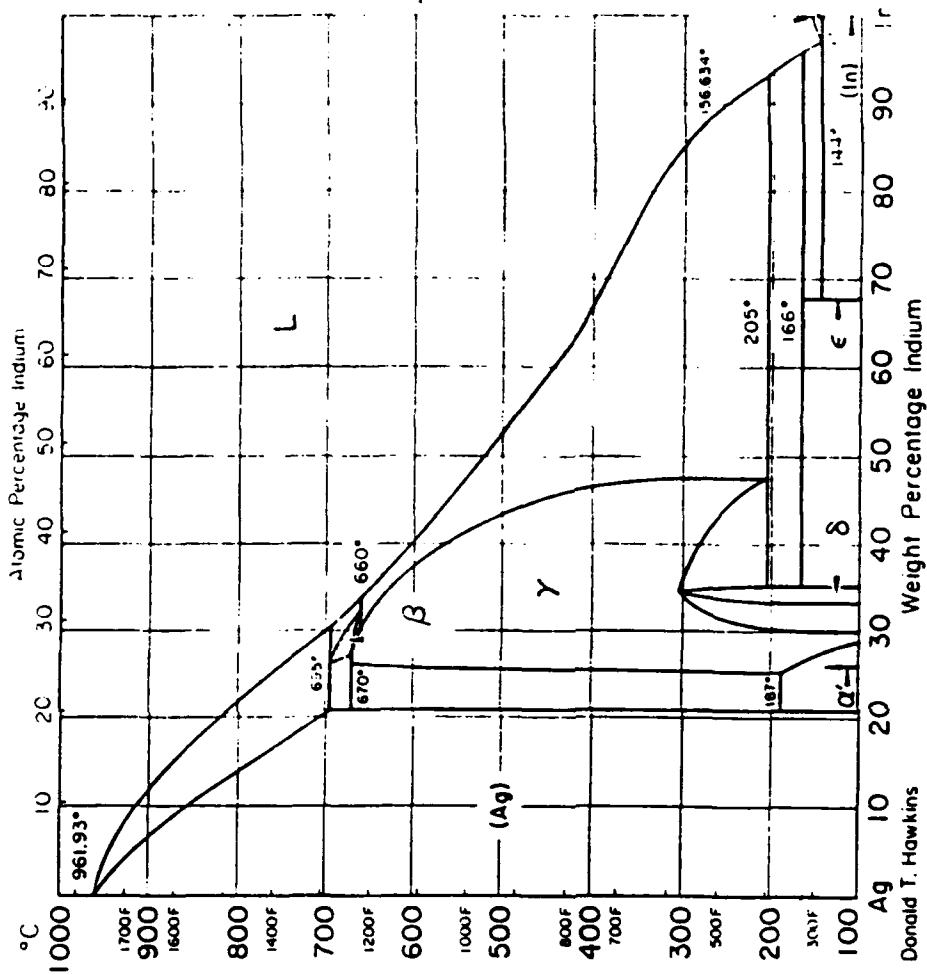


Fig 6

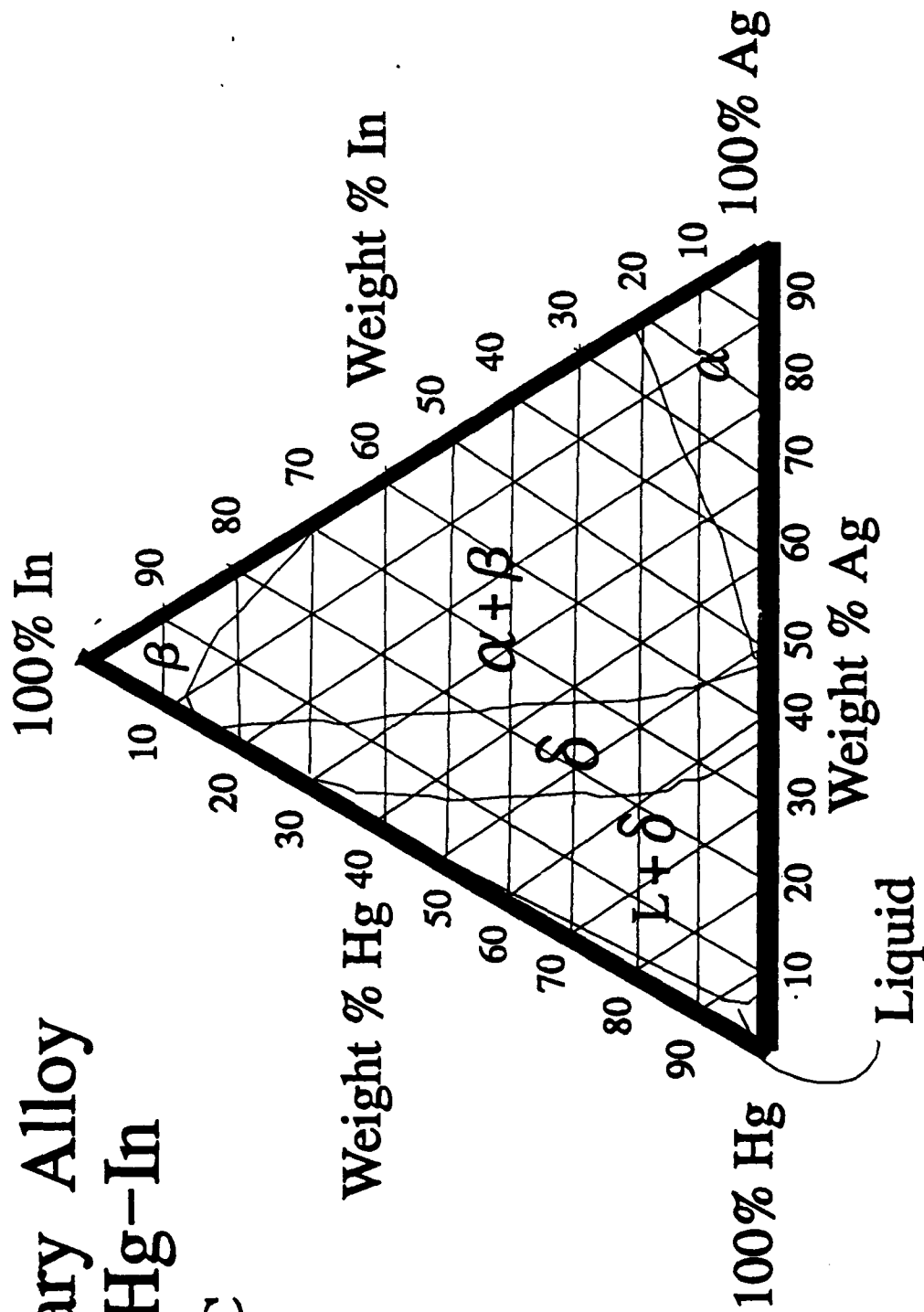
Ag-In Silver-Indium



Donald T. Hawkins

Phase Diagram

Fig. 7 Isotherm of Ternary Alloy Ag-Hg-In at 23 C



Compositions of Samples 1 - Z

Sample	Hg	In	Ag	Uniformity
1	69%	30%	1%	grainy
A	63%	35%	2%	grainy
B	51%	45%	4%	clumped
C	49%	50%	1%	smooth
D	63%	35%	2%	NA
E	80%	18%	2%	clumped
F	83%	14%	3%	clumped
J	73%	26%	1%	grainy
K	82%	16%	2%	grainy
M	83%	16%	1%	smooth
Q	95%	4%	1%	grainy
R	93%	4%	3%	clumped
S	77%	21%	2%	grainy
T	88%	10%	2%	clumped
U	58%	40%	2%	grainy
V	55%	44%	1%	grainy
W	64.5%	35%	.5%	grainy
X	59.5%	40%	.5%	smooth
Y	55.5%	44%	.5%	smooth
Z	50.5%	49%	.5%	smooth
2a	44%	56%	none	smooth

Bulk Electrical Conductivity as computed by the Rule of Mixtures

Key

Ei = electrical
conductivity
in siemens

Di = density

Wi = mass of
element

Mh = gram
molecular weight

ET = bulk
conductivity of
sample

$$ET = E_a \frac{W_a D_a}{M_a^2} + \frac{W_h D_h}{M_h^2} + \frac{W_i D_i}{M_i^2}$$

+

$$\frac{E_h W_h D_h}{M_h^2} \frac{W_a D_a}{M_a^2} + \frac{W_h D_h}{M_h^2} + \frac{W_i D_i}{M_i^2}$$

+

$$\frac{E_i W_i D_i}{M_i} \frac{W_a D_a}{M_a} + \frac{W_h D_h}{M_h} + \frac{W_i D_i}{M_i}$$

Conductivities of Samples 1-2a as computed by the Rule of Mixtures

Sample Conductivity (in siemens)

1	.0691
A	.0849
C	.0917
E	.0584
J	.0639
R	.0630
S	.0705
T	.0369
W	.0685
X	.0732
Y	.0754
Z	.0801
2a	.0817
40% Pb/ 60% Sn solder	.0806

Table 3

**Width of Joints Created with
solder, 232, Z, and 2a**

Solder	.03175 cm
232	.00127 cm
Z	.002032 cm
2a	.00127 cm

measurements taken from Cu-Cu joints

Brian Eplett

Using the IEEE-488 General Purpose Interface Bus
for Device Integration

August 20, 1991

Mentor: Mr. Darryl Huddleston

USING THE IEEE-488 GENERAL PURPOSE INTERFACE BUS FOR DEVICE INTEGRATION

My project for this summer centered around developing software necessary to operate three devices from the computer. This was a challenge of understanding the hardware and applying this knowledge to the software. The hardware knowledge was knowing the National Instrument's GPIB card, the bus, the instruments connected to the card. Lacking experience with remote operations or communications and only limited programming experience, I was to create a program which could run in the MicroSoft Windows environment. Ultimately, my own software would be integrated into to lab work for test and development purposes.

INTRODUCTION

The IEEE-488 General Purpose Interface Bus was originally designed by Hewlett-Packard in order to allow multiple devices to be operated by a single terminal. Unlike most connective systems, the GPIB cables contain both male and female connections allowing multiple interconnected devices. These devices may be daisy chained, laid out in a star-like fashion, or combinations of both. Each device, including the card, can either be a talker, a listener or a controller. In the system used for this project, the controller was the card inside the computer. It is this card which coordinates each of the devices, deciding who listens and who talks. Through the software provided by National Instruments, I was able to configure the card to recognize each device by name. There can be multiple listeners on the bus, but only one device can talk. National Instruments has incorporated into the card an understanding for special commands used to operate the bus. It is with these commands that the programmer uses the system to acquire data.

There are twenty four lines which pass the both bus operating information and computer data, see figure 1. Eight of the lines are dedicated to input/output purposes between devices, including the card. This makes the bus byte serial, bit parallel, eight bits simultaneously, byte by byte. There are eight lines which operate bus

IEEE-488 GPIB Communication Lines

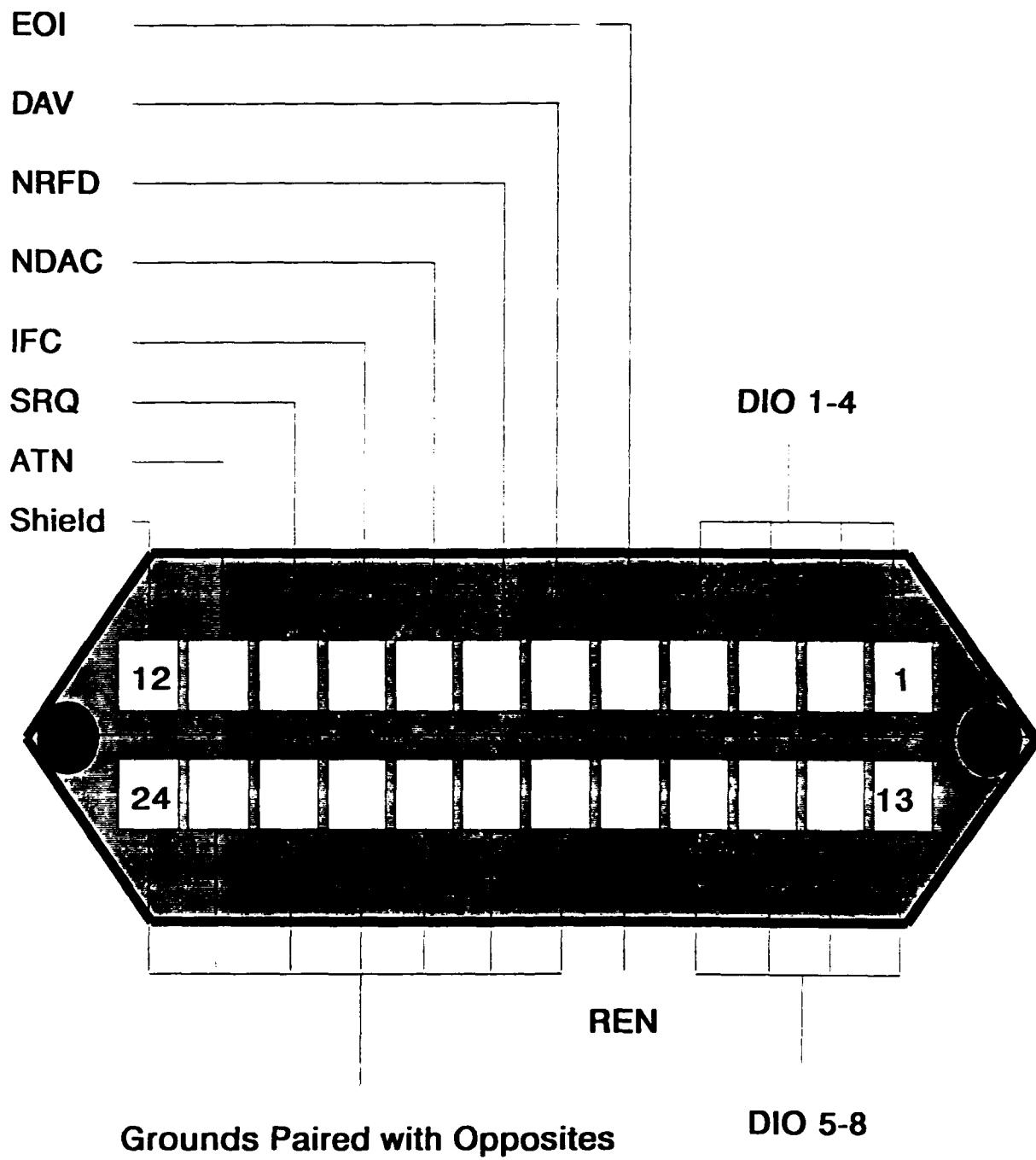


figure 1

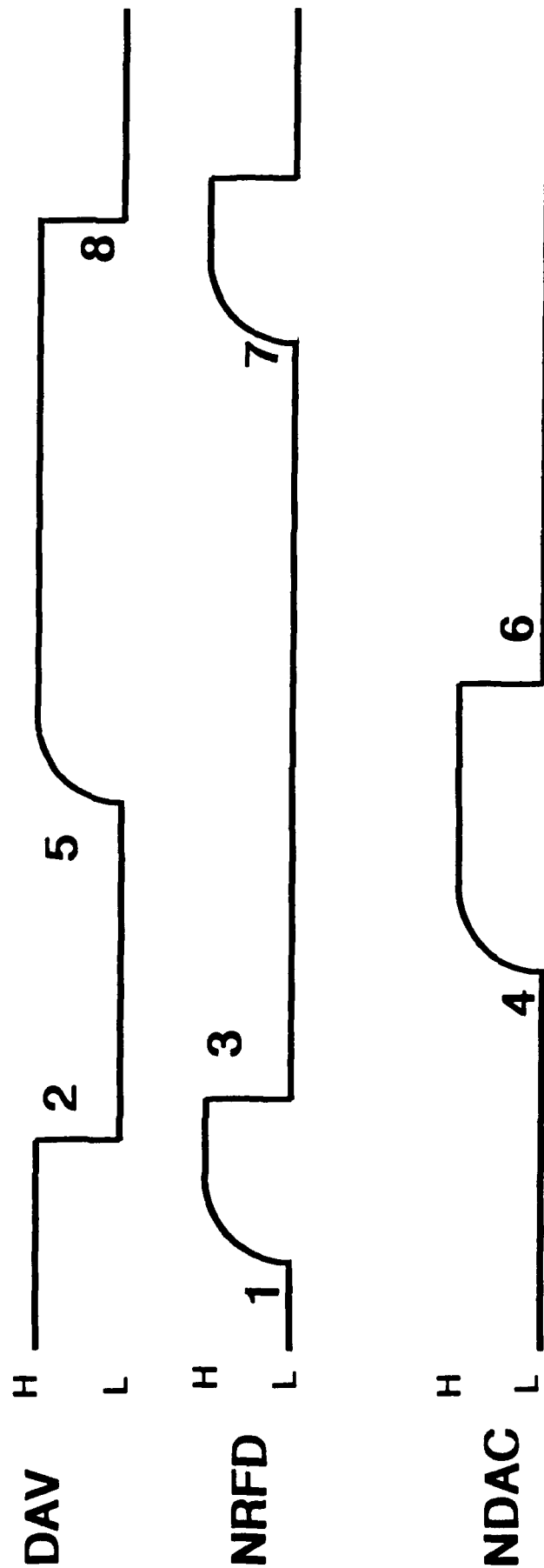
communications. The remaining lines are grounds. Of the eight bus communication lines, three are dedicated to the GPIB handshake routine, and the other five are used to manage the bus. The three handshake lines, DAV, NRFD, NDAC form a critical part of bus activities. In order to transfer data, the bus must verify that the data to be transferred is valid, Data Valid. Then the bus must verify that the listener is ready to accept the data, Not Ready For Data and finally, that the listener has received the data, Not Data Accepted. The management lines establish the conditions of the communication between the talker and the listener, see figure 2 for a description of the handshake routine. ATN, attention, prepares the talker and the listener to transmit and received, respectively, data. IFC, interface clear, clears the bus completely. The SRQ line is the service request line sent by connected instruments if they need the attention of the bus. REN, remote enable, allows the bus to control the actions of the device remotely. And finally, the EOI, end or identify, signifies the end of a transmission. Each one of the lines works in tandem in order to guarantee the passage of data correctly from the talker to the listener.

Another important experience for me was being familiarized with the MicroSoft Windows programming environment. In the Windows environment, I was forced to reconsider many programming concepts in favor of the user

Handshake Protocol for GPIB

DIO 1-8

Data Byte



1. Acceptors become ready for Data
2. Source Validates Data
3. First Acceptor lowers NRFD, no longer ready for data
4. NDAC goes high with slowest Acceptor
5. DAV goes high, data is no longer valid
6. First Acceptor set NDAC low, ready for next cycle
7. Back to 1

figure 2

friendly environment which Windows provided. Less based upon a sequential style the windowing environment provides the user with more control over the operations of the program. After spending many years using an Apple Macintosh computer, it was an enlightening experience to see first hand the process by which many of the applications which I had used were run. The Software Development Kit which was purchased from Microsoft allowed a programmer to create personalized applications for use inside Windows. With this development kit I was then able to manipulate the windowing environment to offer the user freedom within the bounds of the program which I was writing.

Inside MicroSoft Windows, I could incorporate all of these devices and functions to offer the computer user a selection of operations to perform inside the program, or on an image which was in memory. The Windows environment is well suited to allow graphics manipulation, and I sought to develop that ability into my program. With this environment, inside the program which I created, I hoped to incorporate all of these devices in a self explanatory, user friendly, useful manner.

With a slim understanding of the operations of the bus, initial efforts to communicate with the devices hooked up to the card were ineffectual. The first device which I used was the IRIG clock. This clock was attached to a radio receiver which was able to receive an international time

standard being broadcast throughout Eglin Air Force Base. One purpose of operating with the clock was to allow the laboratory to coordinate and synchronize its time with an accepted standard time. Once I felt comfortable enough with the clock, I moved on to the Raytheon TDU- 850 thermal printer. This printer was designed to create high precision grayscale images. This device was an initial glimpse at the tremendous image development available to me. Given any image, I could make a detailed physical picture of it. The final device which I operated with was the Kennedy 9610 tape drive. For this device I had to design a program which would allow the user to operate the drive remotely. This was a critical device to integrate into the laboratory system, for it would allow the addition of data which could only be received through a nine track digital tape and the mass storage of information during laboratory experiments. Inside the Windows environment, I sought to integrate all three of these devices with an image manipulation procedure which could then be a functioning part of the lab.

PROBLEM

The problem which I had to solve was the integration of the aforementioned devices inside the MicroSoft Windows environment, using the General Purpose Interface Bus (GPIB).

In my attempts to communicate with the clock, I began my discover how the bus operates. Because of level of uncertainty involved with the bus at this stage it was a time consuming process to narrow down the problems and discover the errors. Using an example program written in the BASIC programming language by my mentor, and a manual outlining the function of the clock, I was to translate the program into C. Initially, the lack of experience with the bus led to a lack of faith in my program. One of the first problems discovered was the conflict which was occurring between the machine in which the new card had just been installed, and with another machine which had an older version of the same card. Apparently the conflict which resulted from both cards having controller status resulted in a loss of control for both cards and a lockup of the bus. Repeated attempts to run the same program with and without both machines running revealed the conflict on the bus. Even then problems arose. My mentor and I then altered an example program provided by National Instruments to access the clock and were able to properly manipulate the clock via the computer. Attempts then to compare the software which I had written from scratch to the altered software bore little

fruit. I finally noticed that I had failed to include a special header file which was provided by National Instruments which defined the character which signaled the end of the bus transmissions. Through these efforts that we were finally able to uncover the physical problems which the bus was suffering. It was not necessarily the software that was incorrect but the fact that the cables were not properly connected. Only pieces of the messages were being received by the listener because some of the lines were not touching their contacts properly. With the loss of a bus communication line, the bus could no longer perform properly, with the loss of a data bit from one of the data lines, the incorrect information was passed.

Part of writing the new software for the printer was understanding the printer commands, and how the printer would respond to these commands over the bus. An anomaly for the printer was the inclusion of a second bus address. One address was used for commanding the printer, initializing parameters before printing. The second address was used to transfer the actual print data. This data was passed as an intensity character (1 byte) 0-255 from black to white, where each byte represents one dot. This too was a frustrating time for my initial efforts, as each new device presented a whole new list of challenges. Again, the software which I had personally written would not seem to work, so I resorted to adapting an existing example program

to my purposes. It was about this time that my mentor decided that it was necessary to switch my programming efforts to the Windows environment. This necessitated loading an entirely new set of software necessary to operate the special bus functions provided by National Instruments. Now the program which I had just begun to achieve success was useless. My current program could not be integrated properly into the Windows environment because it was designed sequentially. However, I was able to use a Windows environment example program to demonstrate that I could successfully converse with the printer inside windows. Once I was reasonably comfortable with operating the printer from inside Windows, I was given the seed of a Windows program essentially containing the window parameters and the file access routines. From here, I selected the necessary routines from my original program to run the printer from inside Windows. I also included routines to control the size of the image to be printed out, the direction to be printed, or to invert the image when printed. When all the conditions are set, then the image is ready to be printed. As an added convenience to the user as well as a future image development routine, I then wrote the code necessary to display the image on the screen to make the user aware of what image was being printed. By understanding the nature of the data sent to printer after being read off of the disk drive, I wrote a routine which drew, pixel by pixel, the

image on the screen. This was a time consuming process, and the results were unimpressive and inaccurate. I realized that I must set up the colors which Windows was able to use in order to display an accurate picture. This was a new windowing task. In a further exploration of the nature of Windows, I worked my way through the necessary steps to develop a palette of colors on the computer and then use those colors to display the image.

My final task was to join the tape drive into the list of devices available to my program. In order to do this I realized that I would need to change the nature of my program. Initially, I received data off of the disk drive line by line for both printing and display purposes. This was a fairly time consuming process and would be difficult to incorporate into the tape drive process as well. With this in mind, I set out to allocate enough memory to load the entire image into the program to be manipulated from there. But because of the immense size of the files which I needed to be able to load, I could not declare a variable of sufficient size. Instead I had to dynamically allocated 262,144 bytes of memory for each byte read off of the file for the 512 * 512 image. Similar to the task required for using a color Palette in Windows, I had to create the necessary information to allocate the global dynamic memory. At first it seemed that the process was not working, that the system was not allocating the available memory. In

reality it was another quirk of the C language which was causing the system to fail. Efforts to concatenate the string which I had read off of the disk, or even to display the contents of the array were stifled by the fact that black was the equivalent of zero, which is the equivalent of end of string. The computer would input a string where the first character was the end of string character, yet critical image information would still be contained in the middle of the array. Efforts to bypass this unseen error and copy from disk read string to the allocated image array character by character also failed to work. This time it was a data conversion problem. The C compiler causes a specified amount of memory to be allocated for each variable, and for the operations which are performed with those variables. I declared two loop control variables, which controlled the character by character copying, as normal, two byte integers. A two byte integer can span values from -32,767 to 32,767, but the size of the array I was using was 262,144 bytes. Once the loop control variables exceeded a certain point, the integer value returned by the expression which used those values was negative. In effect, the meant that I was requesting memory that was before the array which I had allocated, causing the system to crash. In order to continue to access further memory inside the array, I had to cast the expression and all of its elements to increase the devoted memory and the

maximum number size. Once I had successfully loaded the image onto the image array, I then had to speed the process of displaying the image by putting the image into a bitmap. A bitmap in this sense is simple a "device" specifically created for displaying an image. Like other Windows processes, this required initializing the parameters necessary to displaying the image. Then using a special function which the Software Development Kit provides I was able to directly translate the array into the image. This substantially increased the speed of the display function, and gives an idea of the substantial testing that can be done inside the bounds of the functions which I used on the image.

With these significant Windows processes completed, I was finally able to devote my complete attention toward the tape drive. The tape drive began as something of a mystery. I did not clearly understand the functions and standards which the tape drive used to operate. Tentatively, I set out experiment with these operations discovering how to operate it remotely. Unlike the disk drive where the disk operating system takes care of most of the disk operations, I had to design the search and identification procedures. Again armed with a manual and only a slight example program in BASIC, I began writing a program which merely moved the file position on the drive. This is when I learned how the tape drive recognizes and moves between files. The drive

searches for a specific end of file mark and then stop immediately after it discovers this mark. What this meant is that attempts to keep track of the drive file position eventually were fouled when I changed the direction of the file search. Whenever the drive direction was changed, instead of moving an entire file, it would go to the other side of the end of file mark which it had just passed. This was an easy problem to fix. But they got progressively more difficult. Next, I needed to be able to keep aware of the status of the drive, and even load off the name of the file which the drive was currently on. The status procedure was fairly straight forward, and even helpful to other parts of the program, but the file name presented it own problems. The tape drive does not recognize an end of file mark at the very beginning of the tape, instead it signifies a load point which the drive can never pass. This presented an immense problem. Under special conditions, I had to modify the actions of the drive to be sure it remained at the beginning of the tape. Coinciding with this were my efforts to allow the user to save the image onto the tape drive. When my task was completed the user was able to move around throughout the length of the tape, search for files, know their names, and then access them.

RESULTS

What I have accomplished is an immense programming success. Surrounded by high technology equipment, I had to integrate these devices and allow their easy use. My program will allow the manipulation and visual examination of any $512 * 512$ image. Further, all of the functions which I used to create this program can be extracted and used in other program for similar purposes.

CONCLUSION

This completed the primary goals of my project this summer, but it leaves much of what I learned and experienced out. The true power of my program was its eventual integration into the system of image processing and data collection procedures which are being developed in the Millimeter Wave Lab. One of the potential pieces of hardware which I was introduced to was the DataCube image digitizer. With this device, the lab can digitize and process images real time. My own program would then be used to store or further process these images as necessary. With the printer available to display the image, and the tape drive available to transfer these files, the lab could demonstrate its findings as well as work with other offices toward discovery.

One of the current goals of the laboratory is the Multi-Sensor Instrumentation System. In this system, video, infrared and radar data are all collected and integrated. This data can be used to discover what methods are best for using IR and radar simultaneously and also provide a comparison for other radar and IR systems. My program directly relates to this process. There are two stages to this system, the data collection and the data examination stages during a testing period. At the time which the data is being taken, the video and IR images must be time stamped for comparison, and this can even be extended to the radar

data. In order to accomplish this task, the system must access an accepted standard time via the IRIG clock and place this time on the video and IR images. Further, this information must all be stored in some mass storage device via the tape drive so that it may be examined later. My program and the functions which it includes, can then be used to examine the video and IR images and physically reproduce them.

ACKNOWLEDGMENTS

I would like to express my appreciation to Mr. Darryl Huddleston and Captain Jeff Barnes for their help this summer with getting me on the road to finally completing my project.

HIGH SCHOOL APPRENTICESHIP PROGRAM

Hydrocode Animation

Thomas J. Fraites, Jr.

Niceville High School

Mentor: Mr. Dan Brubaker

16 August 1991

INTRODUCTION

The High School Apprenticeship Program at the Wright Laboratory has many purposes and goals. One of the fundamental aspects of the program is that it enables qualified high school students to perform tasks and conduct research that their mentors do not have time to complete. This mission, I think, has been the basic motivation for my summer research project at WL/MNSA, the Technology Assessment Branch. The animation services that I provided this summer could have been accomplished by any other member of the branch, but no one could spare the time or effort to concentrate on this project. It is for this reason that I have experienced great satisfaction in completing my appointed task. I hope that my work this summer will prove to be beneficial to WL/MNSA for years to come. I myself have gained invaluable academic and practical experience that will definitely benefit me in the future. I feel that this summer I have been given a real taste of life in the science and engineering world.

ACKNOWLEDGEMENTS

I would like to thank the following people for their assistance and support throughout the duration of my time at WL/MNSA:

Mr. Dan Brubaker (WL/MNSA), my mentor, for providing me with all the necessary guidance and background for the successful completion of my summer research project,

Mr. Ken Hutchison (Freeman Mathematical Laboratory), for explaining the workings of the video disc machine, the ultraframe buffer, the VHS recorder, and the system as a whole,

Lt. Col. Robert Donohue (WL/MNSA), for allowing me to work in his branch for the summer,

Mr. Bruce Patterson, Ms. Sharon Joyce, and Mr. David Hogg (WL/MNSA), for their time-saving project hints and suggestions,

Mr. Brian Peterson (WL/MNSA: North Carolina State University) for sharing his workspace and terminal with me for the summer,

Mr. Don Harrison (WL/MNOE), for his assistance with the High School Apprenticeship Program,

And finally, all of the employees at WL/MNSA, for their help, respect, and concern when I needed their assistance.

BACKGROUND

WL/MNSA, the Technology Assessment Branch, has a charter from the Defense Nuclear Agency to assess the lethality of Strategic Defense Initiative kinetic energy concepts against aerospace targets. One of the tools used by WL/MNSA to fulfill this mandate is the CTH Hydrocode. Developed by Sandia National Laboratories, the CTH Hydrocode is designed for multidimensional continuum mechanics analysis, and is used by WL/MNSA to study hypervelocity impact material response. It is a finite difference code, which enables it to calculate and simulate large deformations and strong shocks. In the problems used with the code, materials act mostly hydrodynamically, thus the name "hydrocode". The CTH Hydrocode is capable of dealing with numerous materials per problem and incorporates thermodynamic equations of state (EOS).

WL/MNSA uses the hydrocode for analysis and pre-test predictions. One example of the kind of problems run by MNSA would be the impact of a projectile into a tank of some fluid. The hydrocode prediction would be used to estimate the time and magnitude of the tank's expansion so sensors could be placed in more useful and accurate positions to observe pressure, velocity, hole size, and deformation during a test. The hydrocode is also used to look at problems which cannot be tested using current technology, such as a complex projectile impacting a large mass while traveling at seven kilometers per second. Finally, the code is used to "see" the reaction of materials under unobservable conditions,

such as a reaction that is too fast to observe with available instruments.

The CTH hydrocode does have several limitations. One of the most noticeable of these is that the graphical output of the code is shown in step fashion, that is it shows only single frames. This restriction makes it hard for the user to get a clear image of the entire process that is being simulated by a particular problem. To enhance this graphic capability, a method of animation for the CTH Hydrocode was needed.

My responsibilities at WL/MNSA included the following:

- 1) to refine a method for animation of the single frame outputs of the hydrocode,
- 2) create some useful, animated output of existing, completed problems,
- 3) and to write a clear, easy-to-use manual for use by WL/MNSA, detailing the necessary steps for successful hydrocode animation.

The completion of these three objectives would provide WL/MNSA with products that were previously not available, and the capability to produce them at any time.

DESCRIPTION OF RESEARCH

The course of my summer research project was marked by roughly three different phases: training, procedural analysis, and documentation. These three phases were accompanied by a few periphery tasks (see section entitled "Periphery Tasks"), but for the most part, my summer research was comprised of these three sections.

The first, or training phase, was approximately five weeks long. During this time, I became familiar with the UNIX and UNICOS (Cray) operating systems, learned to use WordStar 5.5, practiced my skills in GEM, and acclimated myself to the different workstations and remote hosts available for my use. These included the Zenith workstation at my desk, the science and engineering VAX processors, the Cray Y-MP, a Tektronix workstation, and a Silicon Graphics terminal. It was also necessary for me to become familiar with the ultraframe buffer machine in the Mathematical Laboratory, so that was included in my orientation period. I also became familiar with several communication programs, as well as All-in-1 on the VAX. Another valuable lesson I learned was the importance of office procedure, both in the military and in any working environment. Our office was very structured, and each individual understood his/her responsibilities. This practical training was as much a part of my summer experience as the scientific and computing education I received.

The second phase of my summer project was the procedural analysis phase. During this time I was required to refine a

rough process for hydrocode animation and to produce a few animated products. This was, for the most part, a trial and error phase. In order to successfully produce some animation, I first had to try different procedural sequences and try to figure out which would best suit my needs. Through experimentation and a little improvisation, a successful procedure was eventually hammered out, enabling me to begin the last phase of my project.

In the documentation phase of my summer project I was asked to recall my experiences in the analysis phase and document the correct procedure for animation in a simple, easy-to-follow user's manual. This task was accomplished, using existing documentation for the hardware devices and my own personal experiences. These resources were compiled and documented, with the final result being the manual that is included in this packet. Hopefully, this guide will be beneficial to the employees of WL/MNSA and other hydrocode users.

DISCUSSION

The process of refining the animation method was a challenging task, and it included several different tasks. To create the animation, my mentor and I knew that we would have to record each frame of the hydrocode output file onto the laser videodisc sequentially, and then use the editing capabilities of the videodisc player and the videocassette recorder to achieve the exact output that we wanted. What we did not know, however, were the particular "keystrokes" if you will, the necessary sequences of button-pushing, to achieve this end. Fortunately, the personnel of the Mathematical Laboratory had already developed a rough outline of the necessary steps. Drawing on the resources and individual experience of Ken Hutchison, we set out upon our chosen task.

The single frame recording segment of the project was somewhat tedious, and often downright boring, but it was a well-appreciated simple task. We simply had to pull up all the frames in order, route them to the ultraframebuffer, and record them onto the laser videodisc one by one. Some of the longer, more intricate problems had several hundred frames, however. This fact, coupled with the varying speed of the Cray Y-MP, could cause this task to take several hours.

The next step was to playback the recorded sequence on the videodisc player, altering the speed to suit our purposes. It was during this step that we learned that for a successful animation project, it is necessary to make the time steps for the

problem output frames as small as feasibly possible. This leads to a smoother, more realistic animated playback.

Once we had achieved the desired speed for the animation playback, we recorded the sequence onto a videocassette. This proved to be the most difficult task. Routing the monitor, videodisc player, and videocassette recorder (VCR) together, and then understanding the intricacies of the editing features of the VCR proved to be very confusing, as well as occasionally frustrating. Once this was achieved, however, CTH hydrocode animation became a tangible reality. WL/MNSA has produced several animation projects to date, including a tape containing five sequences which was sent to the Defense Nuclear Agency to be used to describe the methods and progress of kinetic energy weapon lethality efforts of WL/MNSA.

PERIPHERY TASKS

My time at WL/MNSA was not restricted to preparation and completion of my primary project. During the months that I spent at the Technology Assessment Branch I had the opportunity to assist some of my co-workers with a few smaller tasks.

One of these tasks included retrieving files from Mass Storage at the Air Force Supercomputer Center at Kirtland AFB, New Mexico, and storing them in the Mass Storage facility at Eglin AFB, Florida. This entire task was accomplished without leaving my office workstation. The task was not a short one, however; it took three days to retrieve all of the requested files, valuable time which my co-worker could not afford to devote to that particular task. So I picked up the responsibility during a lull in my own project and saved my branch many hours of what would have been unnecessarily lost time.

Another job that I undertook while at MNSA was the task of helping another co-worker prepare some Vu-Graphs for a briefing. This gave me an opportunity to learn the ins and outs of GEM, a graphics application software package, and provided me with an invaluable tool when it was time for my own briefing. Although this was not as significant a task as the previously mentioned one, my efforts effectively saved my branch many valuable man-hours. When my time at the Technology Assessment Branch was over, I was pleased to know that my time at WL/MNSA had been well-spent and beneficial to the United States Air Force.

HIGH SCHOOL APPRENTICESHIP SUMMER RESEARCH PROJECT

STRESS SENSOR EXPERIMENTATION, ANALYSIS, AND EVALUATION

Daniel R. Grayson

Mentor: David B. Watts

Wright Laboratory

Armament Directorate (WL/MNGI)

Eglin Air Force Base, Florida 32542-5375

INTRODUCTION

This is my first year participating in the High School Apprenticeship program (HSAP) here at Eglin Air Force Base in Wright Laboratory. I'll be attending Choctawhatchee High School this fall as a senior with plans to return to the HSAP program next summer. My summer work project extended from the 10th of June to the 22nd of August during which I learned and contributed in a professional manner to the Armament Directorate. The following material will provide a brief background and description of my research conducted this summer.

ACKNOWLEDGEMENTS

I would like to thank my mentor David Watts for giving me the opportunity and the freedom to fully exercise my intellectual potential during my work project this summer.

I would also like to thank the following persons for assistance and support for my work and for the High School Apprenticeship program:

Joseph Gordon

Thomas Marler

Clyde Lingafeldt

Ken Williamson

Marc Hopkins

Ed Keller

Don Harrison

Lt. Doug Evans

Russel Dukes

Robert Murphey

Rod Powell

Bob Webb

Dr. Klausutis

RDL

BACKGROUND

The main focus of my summer work project this year in the MNGI branch was centered around shock stress sensors. Shock stress sensors measure the force pressure of conventional munitions such as explosives and projectiles. Under the High Explosive Munitions Instrumentation program we are validating the accuracy of these gages. Projects stemming from this program include some data analysis and manipulation, development of software in support of the program, and assistance in actual range tests conducted on those gages and sensors. Two major components are encompassed within the experimentation process. Thin film stress gages utilizing special electrical properties are being compared to a newer sensor technology utilizing fiber optics to measure and determine the explosive potential and effectiveness of developmental weaponry.

Thin film stress gages may be divided into two main types. These are piezoelectric and piezoresistive gages. Although the gages are structurally similar the special electrical properties are quite the opposite. Piezoelectric gages when poled, produce an electrical charge when subjected to stress. Piezoresistive gages exhibit a change in resistance when subjected to stress.

An example of a piezoelectric gage is the PVF2 (polyvinylidene fluoride) gage. It is a thin polymeric film stretched and placed under a magnetic field. This process poles the molecules within

the film. A thin strip of metallic material is metallized onto each side of the gage. The two strands connect at one end of the probe in an area known as the sensing element. When pressure is applied to the sensing element it causes the ploed molecules to give off a charge which flows between the two metal strips on the probe relaying electrical information to a RTD2301 digitizer which provides a graphical representation of the pressure burst. PVF2 gages opetating in the charge mode require a simple resistor and capacitor circuit. The charge generared by the stress gage is trasfered to the capacitor in the external circuit. The charge prodeded by the gage is proportional to the voltage across the capacitor which is, in turn, related to stress. An example of a piezoresistive gage os the Manganin gage. Manganin is a copper-manganese-nickel alloy with a low strain sensitivity, but a relatively high sensitivity to hydrodynamic pressure. An external circuit such as a wheatstone bridge and high voltage source is required for the Manganin gage in order to overcome signal to noise problems in explosive environments. Initially the bridhe circuit is in ballance. However, any change in resistance of the sensor due to stress will unbalance the bridhe, producing a voltage across the output. The bridge output voltage is thus proportional to stress and is linear for small changes in gage resistance.

The other major component encomposed in our experimentation this summer is the fiber optical sensord. Two main types of fiber ptic sensors, the bare wire time of arrival fiber optic sensor and the fiber optic air blast sensor are currently in the experi-mentation stage. The two types of sensors are used for different generl purposes. The bare wire sensor measures the TOA of a shock

wave. The air blast sensor measures the pressure of the force of the outer layer of an explosion.

The bare wire time of arrival sensor measures the time of arrival of a light wave being propagated by an intermediary source which detects a shock wave's movement. It is important for the project engineers to know exactly when in extremely minute time intervals that the source of pressure hit the target material in order to determine detonation velocity. The intermediary step in my experimentation this summer involved a cylindrical structure connected to the target material. Inhabiting the structure was a small crystalline structure that when subjected to a propagated force wave excited and luminesced. The luminescent light given off by the structure was then detected by the bare fiber optic cable giving time of arrival data captured on a digitizer.

The fiber optic air blast sensor determines the pressure at the outer layer of an explosive region. The probe is cylindrically shaped with a conic head. One input and two output fiber optical cables stem from the back of the sensor. A laser light is forced through the input cable and into the probe. A beam splitter within then splits the light wave into two separate components. One component travels unobstructed through the probe and out one of the output cables. The other component travels through a special crystalline structure. When stressed this structure decelerates the light wave component coming out the other output cable. By measuring the difference in velocities between the two beams a pressure may be derived.

DESCRIPTION OF RESEARCH

Initially in order to better understand the digitizer software I was assigned to analyze and manipulate some fiber optic air blast sensor data stored on high density disks. Hydraulics were used to create known pressures. Four shots were taken at each psi reading. Using the TD2301 digitizer software I calibrated the mean values of the offset and baseline of shock wave data. This included obtaining and segmenting waveforms then using mathematical relationships to derive usable values. By normalizing the multiple shots at known psi levels it was possible to create a calibration curve for the polynomial relationships between pressure and the differences in light velocities exiting the fiber optic air blast sensor.

One of the major and most time consuming tasks this summer was the development of software in support of the High Explosive Munitions program. A pressing problem within stress sensor evaluation and its digitizer counterpart is the format in which data points are stored and represented. A time versus voltage format tells little about a munition's strength. The ultimate goal of the program is to determine pressure. Therefore software was necessary to transform the data into a usable form. In order to provide a foreground for future software, I devoted a large portion of my time to the changing the format to a time versus pressure curve. A rather complex algorithm was integrated into some

transformation programs for the MN4-50-EK gage, a piezoresistive sensor. Certain non-static variables for every shot first have to be entered into an equation. The resultant from that function is a transference from voltage to percent change in resistance. Using a calibration curve which compares percent change of resistance to pressure in kilobars, x and y coordinates were obtained. From those coordinates a sixth degree polynomial with a maximum deviation of only .24 percent and an average deviation of .09 percent was derived. By substituting percentage change resistance into the polynomial equation an accurate reading of pressure was outputed.

The most interesting and exciting portion of my summer apprenticeship was the chance to participate and assist in the three range experiments that came up this summer. Calibrations for the fiber optic air blast sensor discussed earlier were taken with Lt. Evans at the Transducer Evaluation Facility. The experimental setup invoked the squeezing of the sensor by a hydraulic press at a sampling of thirty to two hundred psi. Another range experiment took place on the twenty-seventh of July under the supervision of the project engineer, Joseph Gordon, at the C-64-A range. Piezoelectric sensors for determining time of arrival shock wave data were compared to the newer fiber optic time of arrival sensors. A shock wave propagated by comp B explosives was introduced to both a luminescing crystalline structure connected to a bare wire fiber optic sensor that transmits light signals and a piezoelectric sensor that transmits electrical signals. By analyzation of digitized data from the experiment an accurate differentiation between the two sensors was achieved.

Another series of tests were conducted the second week of August at the Advanced Wargead Research Facility. This time under supervision of the project engineer, David Watts, both piezoelectric and piezoresistive gages were subjected to tests. Explosives were mounted to target material in which the stress sensors were located. A substantial data base of digitized signals from those experiments was obtained in the nine successful shots.

BIBLIOGRAPHY

Digital Research "Gem Desktop Publisher" User's manual 1988.

Microsoft "Learning to Use Microsoft QuickBasic" User's manual
1988.

Microsoft "Programming in BASIC" User's manual 1988.

Tektronix "TD2301 Transient Digitizer System" User's manual 1988.

Wordstar International "Wordstar References" Reference guide 1990.

1991 HIGH SCHOOL APPRENTICESHIP PROGRAM
FINAL REPORT: TRANSFORMS, IMAGE COMPRESSION

Derek Holland, WL/MNSI



**1991 HIGH SCHOOL APPRENTICESHIP PROGRAM
FINAL REPORT: TRANSFORMS, IMAGE COMPRESSION**

Derek Holland, WL/MNSI

ABSTRACT:

This summer I participated in a variety of different projects, all 'housed' under the giant arms of Signal Processing. Working in the Guided Interceptor Technology Branch, Strategic Defense Division, I wrote several series of programs with varied applications and uses for my office, working generally with my fellow apprentice and colleague, Troy Trquhart, and the established computer guru of our office, Capt. Allen Andrews. My mentor, Paul McCauley, helped in teaching me of the 'real world,' some bizarre place he says exists beyond college (I think he's a bit crazy). This year the majority of my work can be categorized as dealing generally with transformations of images - thus the reference in my title. It was then a refreshing discovery to realize that this newly-acquired knowledge of image frequency spectra (used primarily for signal processing operations) could be used to shrink the storage size of image data - the Discrete Cosine Transform and Wavelet (fractal) Transform Compression schemes.

INTRODUCTION:

First, to clear up any misconceptions as to exactly WHERE WL/MNSI 'fits' into the grand scheme of the Department of Defense, I shall describe the functions of my office. Under the Wright Laboratories (WL) section of DOD, subordinate to the Strategic Defense Initiative, MNSI is involved with

research relating to Guided Interceptor Technology (here I refer to a purely kinetic weapon that is space-based).

Before we continue, permit me to explain exactly what I mean when I say "I programmed." During the last few summers I have had the opportunity to work on several SUN graphics workstations, multitudes of Digital Systems' VAX computers, an IRIS 4d Silicon Graphics Workstation, and numerous personal computers. During that time, I have learned C, FORTRAN, UNIX, PCL, some Assembly Language, and, in general, a deeper understanding of electronics (I also gained a powerful notion of why many professional programmers do not use BASIC for their algorithm development).

The organization of this report is similar to the organization of my summer: I had no ONE gigantic project that took my time. Therefore, I shall approach each task separately, drawing overall conclusions toward the end.

TRANSFORM IMAGE FILTERING:

By definition, a transformation is simply a way of repositioning data - "mapping" it, if you will - into a format labeled as "easier to process." When one speaks of the natural order of image distribution, he speaks of a "spatial" representation - that is, equal to the time/space relation that our eyes perceive. This format, though in most cases the most esthetically pleasing, is not suited to image processing application. For instance, if one wishes to enhance the image - to remove noise in a particular region, he is limited by the fact that this noise can be in ANY pixel location within the two dimensions x and y. Garbage does not necessarily conform to one's wishes (remaining in only one region of the image) while still in a time domain. However, since most garbage in a set of data will exist at certain

frequencies. a transform, redrawing the image from high to low frequency, will enable faster and easier processing. The Fourier Transform accomplishes just this task, producing a frequency spectrum of the image to be enhanced. Now, extraneous data is more correlated in frequency (one loses an x/y correlation), providing a more convenient representation for the above 'cleaning' operation.

Thus, if the spatial image $f(x,y)$ is continuous and integrable, and its Fourier Transform $F(u,v)$ is integrable we have the following two dimensional transform and its inverse:

$$\mathcal{T}\{f(x,y)\} = F(u,v) = \iint_{-\infty}^{\infty} f(x,y) \exp[-j2\pi(ux + vy)] dx dy$$

and:

$$\mathcal{T}^{-1}\{F(u,v)\} = f(x,y) = \iint_{-\infty}^{\infty} F(u,v) \exp[j2\pi(ux + vy)] dx dy$$

It follows, then, that the brunt of my work (and also the most obvious section) involved reproducing the Fourier on the SUN workstation - the SUN was already equipped with an extremely powerful imaging utility, SAOimage (NOTE: by this I mean more powerful than I could possibly sit down and begin to write). So then, the difficulty lay in translation of mathematics to computer code. Fortunately for me, I had the services of a great bunch of people, who succeeded quite thoroughly (and may I add utterly completely) in confusing and baffling me beyond that which I could have possibly done on my own. I also thank the author of **Numerical Recipes in C** for adding to my despair. It was

at this time. I might add, that I made my discovery into the truth of the Fourier Transform: put simply, that NO ONE REALLY knows anything about it. It was a great help to me when I received fifteen DIFFERENT explanations as to the derivation of the equations (NOTE: it is my complete theory that to graduate, all engineers must submit an ORIGINAL explanation as to why the Fourier works).

After a few days, though, I managed to write a working version of the Fourier program that transformed an image (a black box), and then performed a similar inverse operation. The program was written in C, and was tested with another program that I wrote, called 'box.c' which drew you pleased it, a box, the spectrum of which (a point) was known.

The next step in the project was to write 'chop' programs, so that one could transform parts of an image - to be reassembled at a later time. The image used for this series of programs was a digitized view of the Statue of Liberty. Basically, the execution of the programs involved taking the original image and cutting out the center 128 square pixels (the original image was 512x512 pixels). The FFT (Fast Fourier Transform) was then taken of that subregion, and then a splicer program reassembled the two images. Now, a user could take the Fourier Transform of a particular region, and thus compare the spectra of two subimages.

The final step of this project was the design of the sharpening utilities - the operations performed AFTER the image was transformed. The program I wrote is called a high-pass filtering system, in that only the higher frequency data is released to the final picture (up to a user-specified point). This has the effect of 'lightening up' any picture too scrambled, noisy, or dark to see detailed information.

To explain more intricately the details of the filter, one must remember the data of the image arranged from high to low frequency. If this information is graphed on the x/y plane, one sees a steady slope downward - a curve starting high (the peak intensity pixel) and then bottoming out. The filter simply passes only the higher frequency data, the cut-off chosen by the user. This is performed by first computing relative power-spectra cut-off points in the data, and then figuring their importance to the total amount of power present in the image. For example, if the user wishes to accentuate the top ten percent of power in an image, he runs a finder subroutine which, in effect, draws circles. Remembering the 'Fouriered' image, one sees all data centered on a point toward the middle of the display. By drawing concentric circles of various radii, then, one finds various percentages of total power. Back to our example, though, once this 'circling' finishes, the user receives a printout of the results of the radii. The choice is simple. The user selects the circle encapsulating a percentage closest to his/her original estimate (for example, 11.4 percent verses the original guess of ten percent). The program runs with a **BO** value of this number as an effective boundary for the equation. For all curious, the equation is defined as the following:

$$H(u,v) = \frac{1}{\{1 + .414[D_0/D(u,v)]\}^{2N}}$$

Thus, the image is enhanced by using the Fourier Transform - to establish a format easily processed.

TRANSFORM COMPRESSION:

Because image transforms reduced a set of data to a frequency map, MNSI was interested in experimenting with a transformation-based compression system. Due to an inefficient telemetry unit in the interceptor, data at present could only be transmitted at the loss of nine frames for every ten. To attempt a solution to this problem, I was eventually joined by a small team from our office (in this, my previous experience with the Fourier helped). Since most compression algorithms are divided into a mapper, a quantizer, and an encoder, I shall organize this section accordingly.

MAPPER (TRANSFORM)

The mapper simply reformats one set of data into something more manageable with which to work. There are incredible amounts of work available concerning various different compression schemes, but the one we chose to adopt initially was a Discrete Cosine Transform "based" compression scheme. When studying the effect of transforms upon images, we realized that one of the objectives of the mapper was to reduce the amount of correlation on the new spatial plane. Since the Cosine Transform is derived directly from the Fourier Transform (basically, the real component), it satisfied all requirements for general translation. Although we made tiny modifications in the approach, the basic equation for the transform (which was, by the way, coded by Troy McQuharty) is given by:

$$C(u,v) = \frac{1}{2N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) [\cos(2x+1)u\pi] [\cos(2y+1)v\pi]$$

(INVERSE SWITCHES $C(u,v)$ WITH $F(x,y)$ + NORMALIZES)

There were several details still to investigate, however. Initial trial runs found that a simple two dimensional DCT was too computationally intensive

to be used. The DCT, as one sees above, involves a hefty amount of summation. More, in fact, than one can deal with real-time on a 118x128 focal plane image.

And so, like the HEAF's we use, Tony and I moved on, regarding the DCT industry standard. Changing strategy a little, we adapted a block quantization method (more on this later), where an image was subdivided into $M \times N$ (for our purposes 64×8) regions, the DCT taken on each subimage. This method had, to its advantage, a great deal of correlation between pixels in the same space relative to the 64×8 grid. It was our belief that this format required less information to store the data.

This method of transformation, however, proved again inefficient. We were joined at this time by a fellow colleague, Capt. Allen Andrews, USAF, who suggested some interesting modifications (which he implemented) that enabled the DCT to work. He stayed with the block quantization, but, instead of using a mathematically costly two dimensional approach, tried a quicker one dimensional transformation. The subimages then became ROWS in the image, arranged from high to low frequency horizontally. The three of us then realized that in that format, there was a high degree of correlation VERTICALLY, since the pixels were grouped in similar rows. Thus, by using a technique known as Differential Predictive Coding (DPCM), we were able to further the potential for reduced storage space. DPCM, in its simpler forms is basically a subtraction. You see, we knew that if one found differences pixel to pixel in the vertical direction, the resulting deltas tended to be extremely small - so small, as to theoretically require an insignificant amount of information to code. The difference (and its respective storage

size) became even smaller when we adopted a linear predictor. This tiny subroutine basically 'prophesied' the next pixel's value, with a higher degree of accuracy than mindless subtraction - the deltas averaged 0's to the thousandth decimal place. Needless to say, this compromise-hybrid method was the one we adopted.

BLOCK QUANTIZATION

After an image has been transformed (and in later cases differenced), one needs a routine to decide how many bits to 'spend' on each pixel of the image. Contrary to public opinion, the Cosine Transform does not compress - it, if anything, merely expands the range of the image. The actual compression is in itself lossy - a quantization of sorts.

By definition, when one 'quantizes,' he pockets the data, in its existing range, in the desired number of bits (example: 3 bits, 8 grey levels, $2^{(number\ of\ bits)}$). This tells the encoder the information it desires. The problem, though, still exists: how does one find the appropriate number of bits necessary for image encoding, still preventing irrevocable data loss?

The answer lies in variable bit lengths. Using the principle that more 'important' pixels tend to possess greater pixel-to-pixel variance over a frequency spectrum, one can calculate a bitmap for the one-dimensional 'rows' of our image. We used the following log-variance equation to make assignments:

$$N(u,v) = \frac{N_0}{N_z} + 2 \log_{10} [V_F(u,v)] - \frac{2}{N^2} \sum_{a=1}^N \sum_{b=1}^N \log_{10} [V_F(a,b)]$$

Originally, when Troy and I calculated the bitmaps for 8x8 sub-regions,

it was necessary to find a pseudo-average to transmit. With the one-dimensional solution, however, such redundancy became unnecessary. One could simply hardcode an average bitmap into the quantization process itself, for once the map WAS found, the operations became very simple. To summarize, the programs then went through and found ranges for all pixels encoded with equal numbers of bits. Each bit assignment was then coded with a number corresponding to its place in the respective bit map, be it a level of two out of sixteen, or four out of a total two hundred fifty six. This served as the main force of compression.

IMAGE ENCODING

The final steps of compression existed as the bit-reads and writes - the subroutines that actually did the coding of the data as a smaller set. For this part of the scheme, Capt. Andrews and Troy did the majority of the programming - I served mainly for ideas and debugging. Basically, our encoding approach amounted to several subroutines - one that wrote data to a device bit-by-bit, and another that read it back.

COMPRESSION RESULTS

After finishing the coding, we found optimum compression ratios at approximately 8:1. Any attempt at further compression resulted in data too scrambled to process. We also found an interesting phenomenon - that naturally created objects (digitized pictures) compressed extremely well, while artificial objects (our simulated plumes) did not fare as highly, becoming well organized garbage after minimum compression. Our theory is that

this relates the intense frequency packeting that occurs in a DCT (an artificial image reverses the gradual down-slope of a natural object). When one loses any information out of one of the 'packets' damage is irreparable. In all, the project was a success as to a possible way of solving the telemetry dilemma, at least until further, more advanced technology becomes practical.

FRACTAL COMPRESSION

I had an opportunity to study two methods of fractal compression to compete with the DCT to solve our telemetry problem. The original object here was to compare as many compression methods as possible, selecting an optimum method as the end. To prepare for these assignments, I studied fractal math for a good part of the summer, writing various fractal-drawing programs that ran on the SUN.

One method, formulated by Dr. Michael Barnsley, uses a technique known as IFS coding, where an image is broken up into self-similar images (hundreds or thousands) that are represented in terms of coefficients. These Iterated Function Systems (IFS) codes, then, can be used to restore the original picture. This process can achieve a tremendous compression ratio, but faces the arduous transformation of image to fractal. Dr. Barnsley has now somehow found a way to find the inherent self-similarity of any image, and then store it as so many affine transformations. This method, however, was based entirely in hardware (preventing me from studying algorithm technology), and thus useless. Barnsley is releasing nothing until he is totally 'patentally' secure.

The other method, used a wavelet function to compress, breaking an image

into primitives, and then storing results of different scalings as 'peaks.' Research, though, was based in the UK, arriving too late for me to begin experimentation. To present any other explanation of these two complicated theories in this report would be foolish and FOOLISH (did I mention the foolishness involved?).

OVERALL SUMMARY

The compression work goes on, even as I write this. The technology here is incredible, and IN SPITE of that, my methods worked. I gained an incredible amount of knowledge during my stay at MNSI.

I define my last summer with the HSAP program as an incredible success. I participated in actual crisis-level projects, and had the opportunity to contribute my ideas as an equal. I can ask for no more, and thus thank all involved for an incredible summer (and for that matter, an incredible three years!). To WL/MNSI, and to SDI, I wish full success.

Bibliography

My most important sources this year were human, not paper and cardboard. Therefore, this section is largely unnecessary. Books had a weird habit of confusing me, and since most of my work was original, I refuse to include a gigantic Bibliography page.

Gonzalez, Rafael C. and Wintz, Paul. Digital Image

Processing. Massachusetts: Addison Wesley, 1987.

Pratt, William K. Digital Image Processing. New York:

Wiley and Sons, 1978.

Acknowledgements

I would now like to thank the following people, to which I am indebted, and without which, I could never have really accomplished anything:

Capt. Allen Andrews:: The unofficial office guru for all computer-related 'stuff.' Thanks for the advice.

Mike Douvillon:: For never becoming impatient with my computer use or practices.

Victoria Franques:: For calling me 'opaque' and crashing my SUN.

Dennis Garbo:: For letting me become better acquainted with the Silicon Graphics machine, and for being a great Fourier teacher.

Bob LeBeau:: For letting me date his daughter Danielle (in about 16 years).

Paul McCarley:: For being the ultimate mentor, and having a groovy music collection. Jam on, Paul.

Alice McCrae:: For always remembering my name, and locating me.

Lee Murren:: For always speaking his mind - never mincing words (for some great stories, too).

Mickey Phipps:: For the incredible SPPD class we 'attended.'

Bob Stockbridge:: For the PC card experiences.

Troy Urquhart:: For every success I enjoyed. Take care.

Greg VanWiggeren:: For the good times. God bless you, wherever you are.

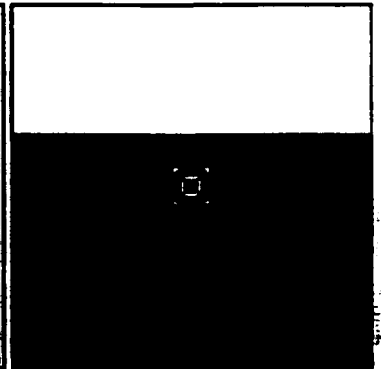
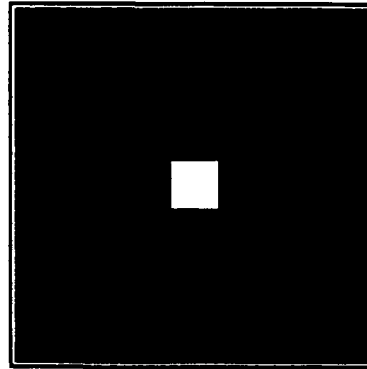
Dorothy Williams:: For allowing me the great honor of assisting her in image enhancement 'studies.'

I wish to thank everybody else in MNSI, and anyone else left off (I assure you, it was accidental). A memory if a frail thing, and mine is

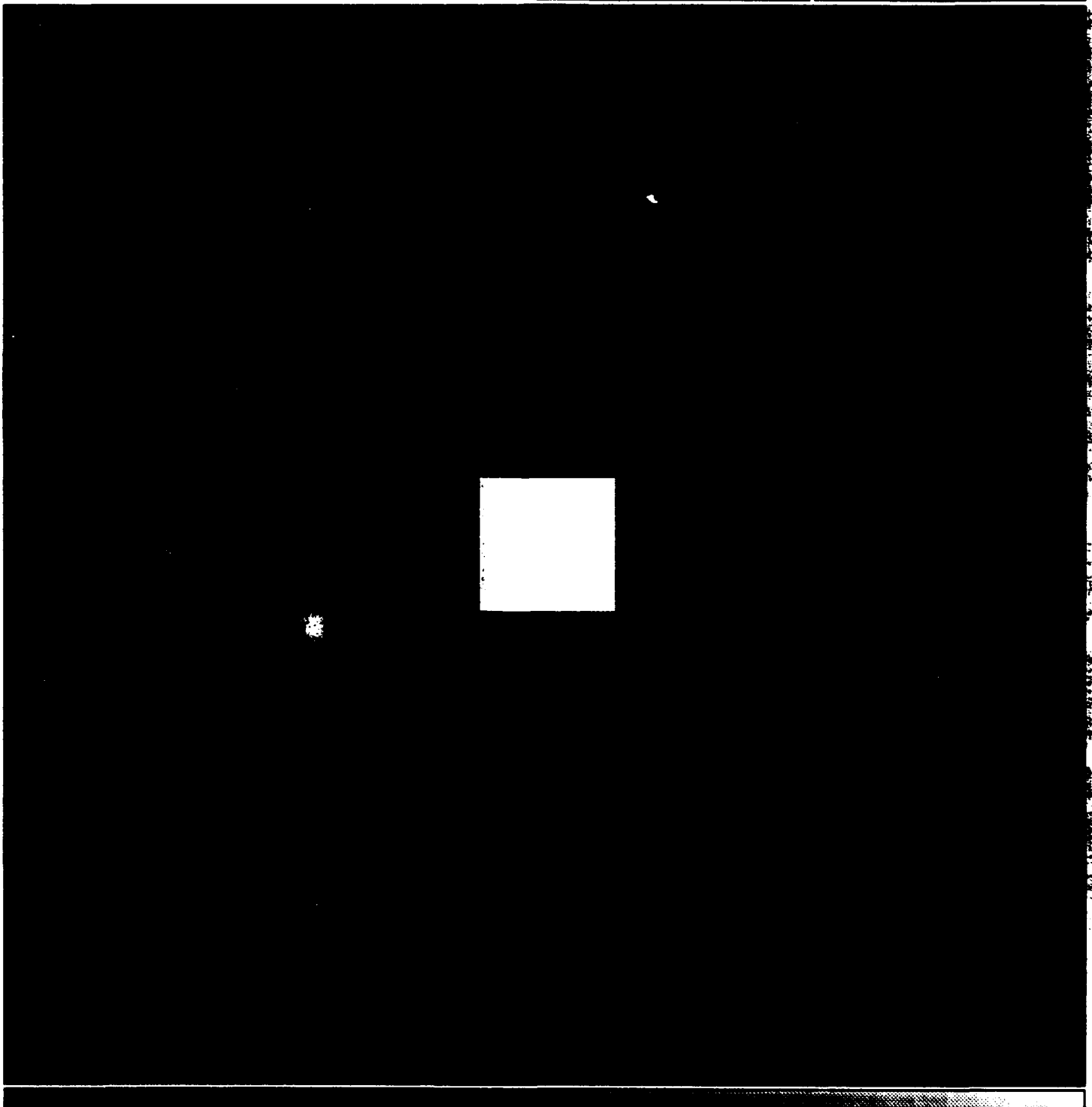
bedridden. Finally, I'd like to thank Mr. Don Harrison, for leading a great program, and RDL for making it possible.

GO RICE OWLS!

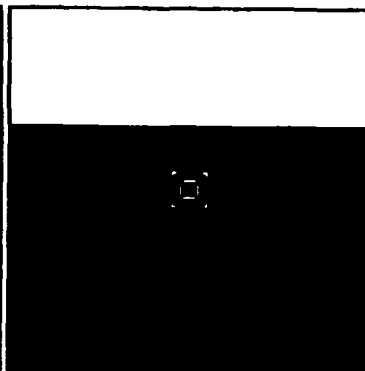
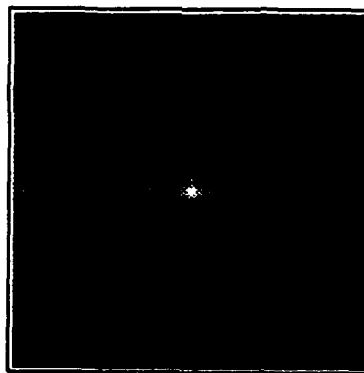
file: picture.img
dir: .



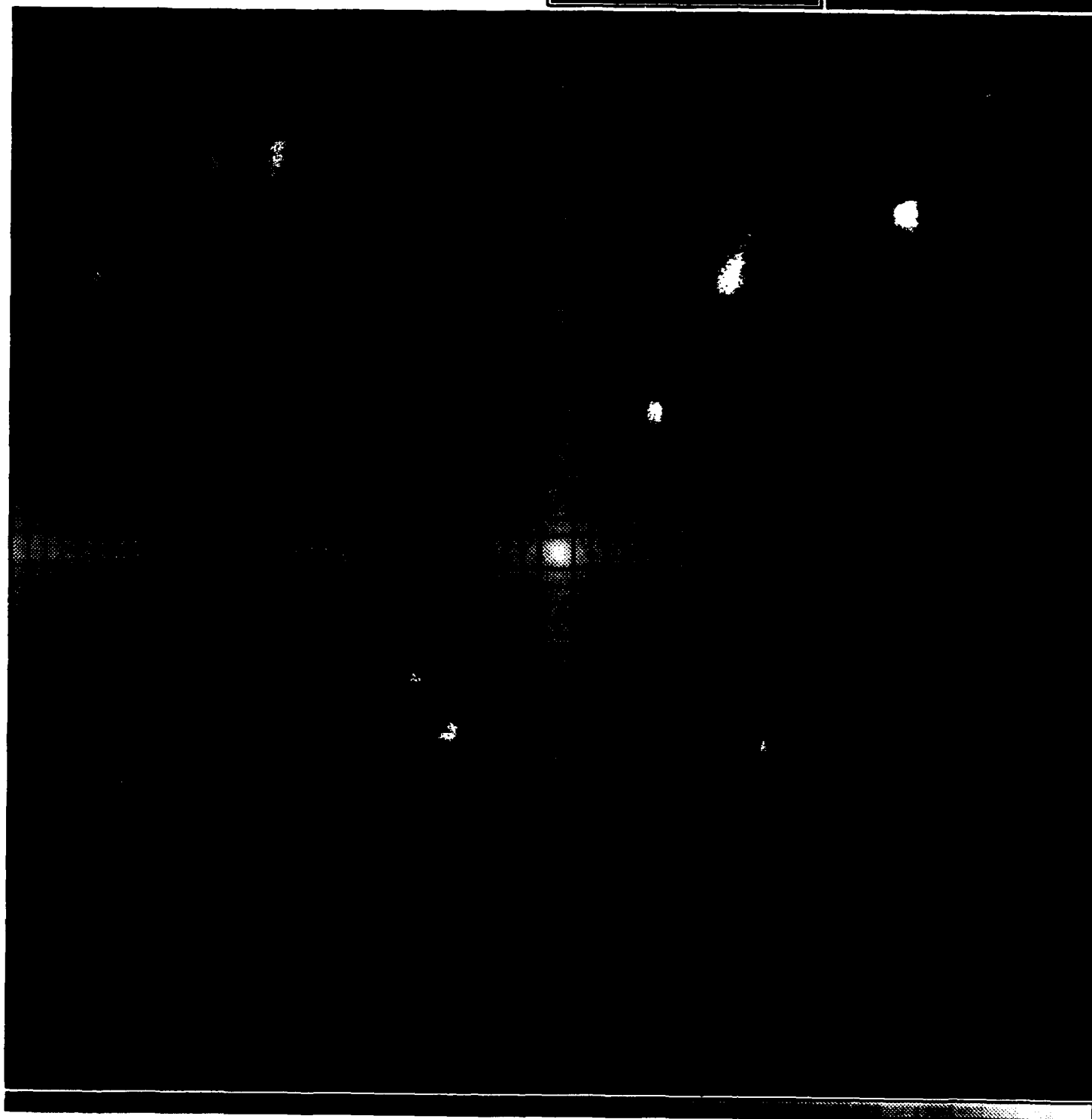
229.0 508.0 0



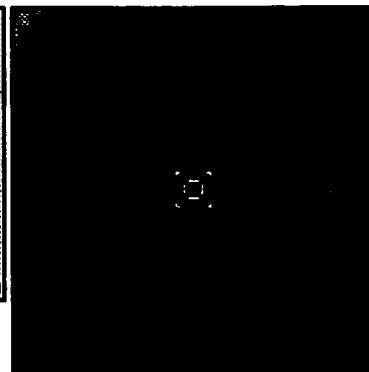
file: picture.out
dir: .



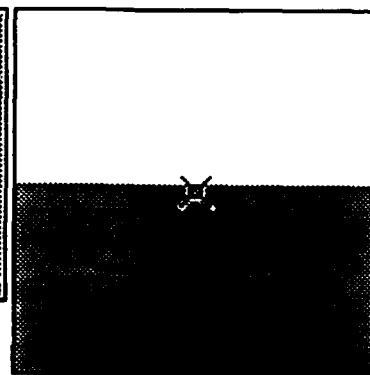
451.0 507.0 1.895e+07



file: liberty.img
dir: .



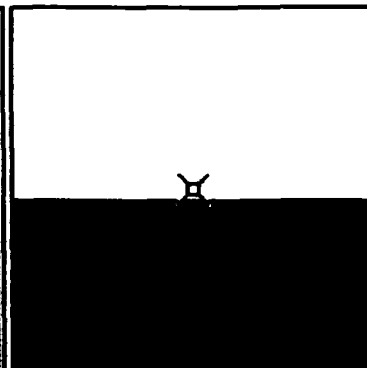
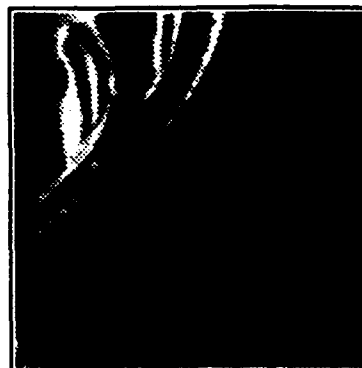
file: liberty.bak
dir: .



93.0 1.0 122



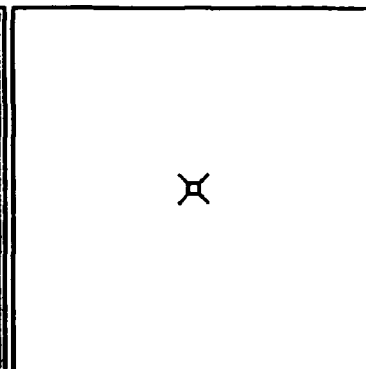
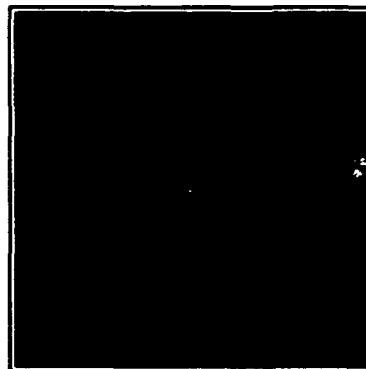
file: picture.img
dir: .



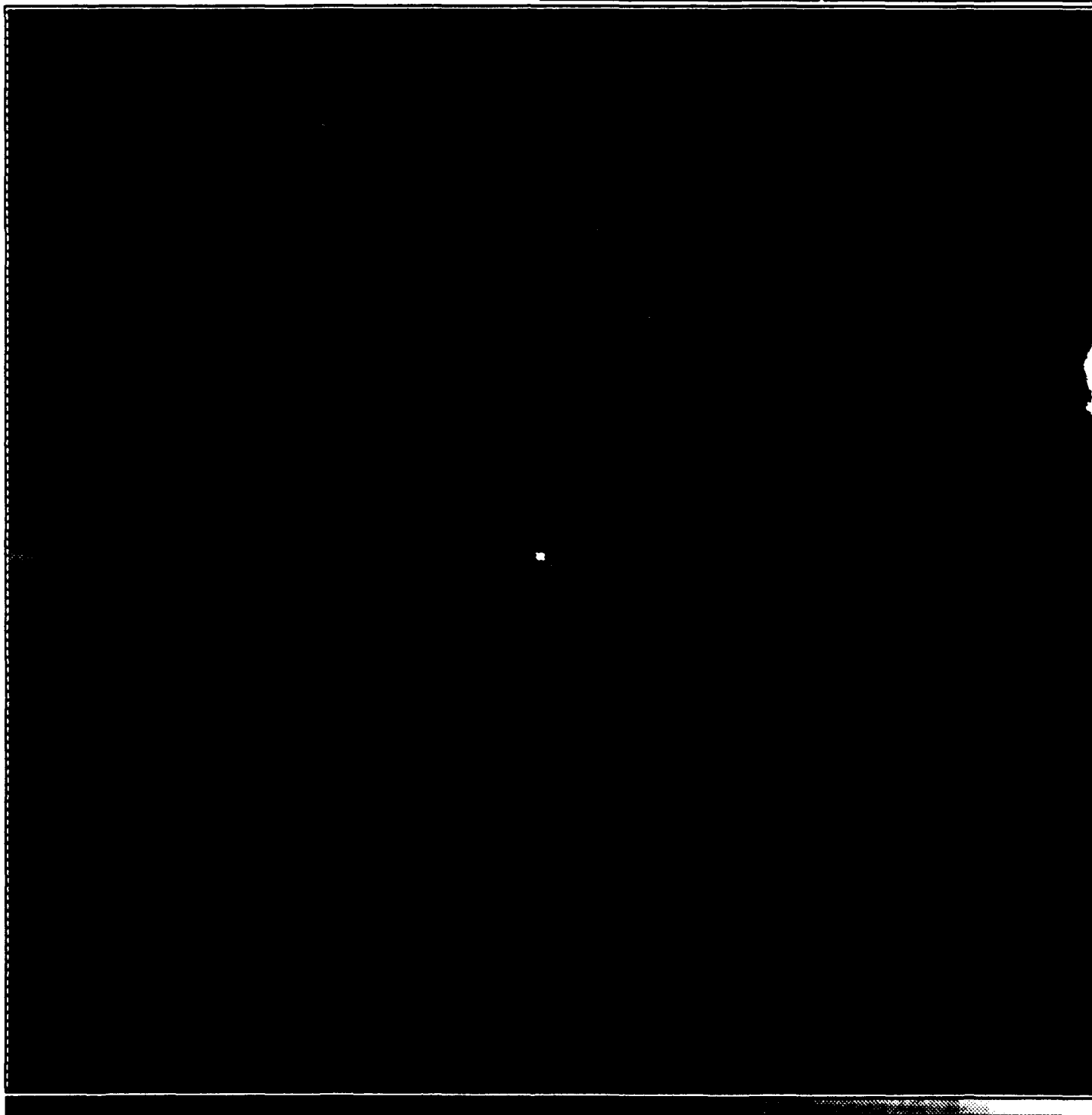
45.2 0.2



file: picture.out
dir: .



81.0 -185.0



CHAD HOUGHTON

MENTOR: DR.DENNIS
GOLDSTEIN

HSAP RESEARCH: LASER
POLARIMETRY

21 AUG 1991

VISIBLE LASER POLARIMETRY

by:

Chad Houghton

INTRODUCTION

During my first summer in the HSAP program, I worked in the field of laser polarimetry. Research in laser polarimetry has progressed from the early stages of development to advanced measurement and collection used to test various samples. The infrared laser polarimeter was first assembled by Dr. Goldstein and Mr. David Chenault. The first measurements and a description of the polarimeter can be found in Mr. Chenault's final report.¹ Mr. Randall Hodgson did additional work on the polarimeter which included steps to reduce measurement and data processing errors; his work is documented in his final report.² Follow up research, data collection and calibration of the polarimeter was completed and compiled by Mr. Randy Gove and documented in his final report.³ Additional information on polarimetry or electrooptic modulator materials can be found in Dr. Goldstein's dissertation.⁴

My part in the research of laser polarimetry at Wright Laboratory dealt with the calibration of the carbon dioxide infrared polarimeter and collection of data involving wavelength and power

output, as well as the building of, calibration of, and data collection from the HeNe visible polarimeter. Various samples were measured with both polarimeters to determine the polarization properties of the materials. Most of my research this summer pertained to the HeNe visible polarimeter which I constructed. Both polarimeters are designed to test different materials to see if they would be a potentially good electrooptic modulator. Improvements made on these electrooptic modulators would enhance target simulation systems, optical processing systems, image processing systems, and eventually optical computing.

I am currently attending Choctawhatchee H.S. and will be a senior in the fall. This is my first year in the High School Apprenticeship Program and I plan to continue my work at the lab. I hope to be able to further my knowledge and understanding of the field of laser polarimetry by returning next summer and extending my research capabilities.

ACKNOWLEDGMENTS

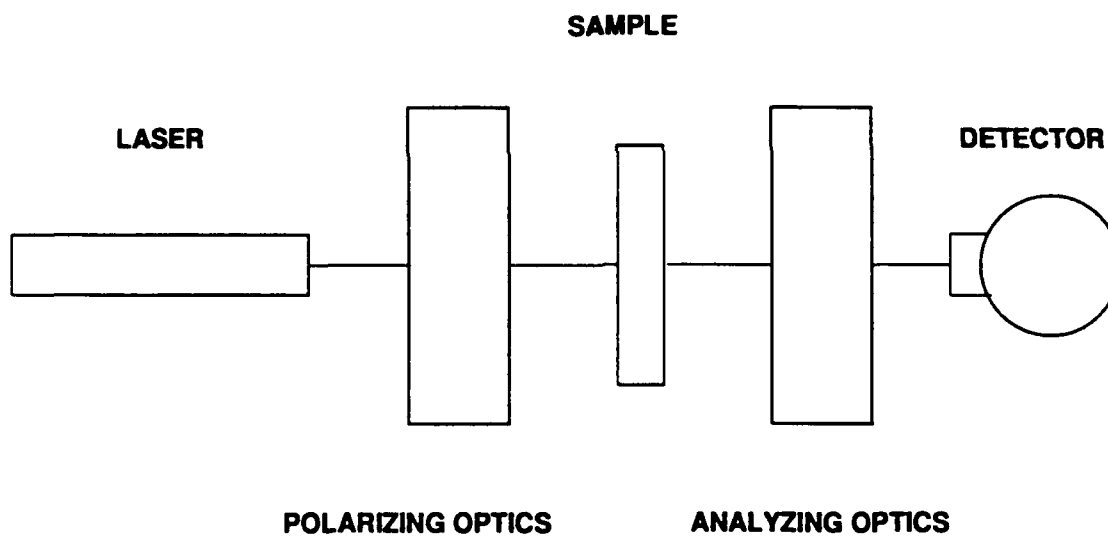
I would like to acknowledge the Air Force Armament Directorate for allowing the High School Apprenticeship Program to exist and for providing valuable research opportunity as well as opening many doors to young people like myself. I would like to thank the Air Force Office of Scientific Research for funding the program. Research and Development Laboratories must also be thanked for their excellent administration of the program. My most sincere gratitude is extended to my mentor, Dr. Dennis Goldstein, for taking the time to guide my progress and for placing his trust in me to complete work on the polarimeters. I would also like to thank the technicians in the Special Projects Laboratory for their laughter and solutions to our many problems: Voncile Houston, Michael VanTassel, Howard McCormick, and Linda Lau.

My deepest appreciation goes to Randy Gove who helped me every step of the way and taught me more than I'd ever thought I could have learned in one summer; his patience and understanding were the greatest lessons I learned, and allowed me to not only experience the research but to grow from my experiences. Finally, thanks and gratitude is sent to Annette Marsh, Lisa Collins, Danielle Walker, and Dr. Charlesworth Martin for their warm friendship and for making me feel like a part of a large family that never stops enjoying life.

BACKGROUND

A polarimeter is an optical instrument used to test polarizing and retarding properties of light and different materials. If the behavior of the polarimeter on the light is given knowledge then the polarization change produced by inserting a sample into the polarimeter may be determined. The visible polarimeter uses the Mueller matrix polarization formulation to process and express results.

Figure 1 shows a block diagram of the polarimeter.



D. Goldstein, 12 April 1980

FIGURE 1

The system can be divided into five different sections: the source, the polarizing optics, the sample area, the analyzing optics, and the detector. Figure 2 depicts the actual setup of the polarimeter.

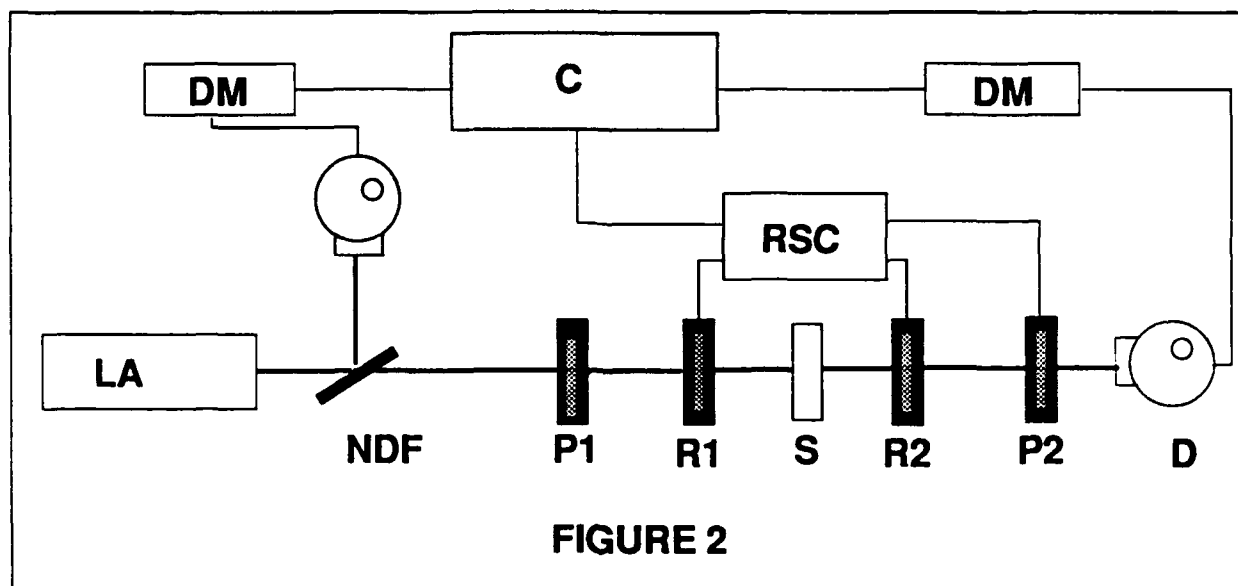


FIGURE 2

LA	LASER
D	DETECTOR
P1,P2	POLARIZERS
R1,R2	RETARDERS
S	SAMPLE
NDF	NEUTRAL DENSITY FILTER
C	COMPUTER
RSC	ROTARY STAGE CONTROLLER
DM	DIGITAL MULTIMETER

The source is the HeNe laser mentioned before. The polarizing optics consists of a polarizer mounted in a manually controlled rotation stage and a quarter wave retarder mounted in a computer controlled rotation stage. Because the light coming out of the laser is already polarized, the first polarizer is ideally unnecessary. However, preliminary observations and tests proved that the laser light was only partially linearly polarized. The sample mount is followed by the analyzing optics which consists of another quarter wave retarder and another polarizer both mounted in computer controlled rotation stages. Two detectors were used so data could be taken

on a sample from one detector while the other detector monitored the laser. I accomplished this by installing a neutral density filter set at a forty-five degree angle. The filter not only sent the beam through the analyzing and polarizing optics to the second detector but also attenuated it into the first detector which kept track of the laser stability. One important reason for using the filter was to prevent the beam from saturating the detectors. One of my tasks involved in the construction and calibration of the laser was to rewrite the stability program from the infrared polarimeter to work on the visible polarimeter as well. The stability program records a given number of intensities from the laser at a particular time interval and compares them in a line graph. The stability subroutine in the Mueller matrix program allowed us to make calculations and monitor laser stability on the same sample or calibration run.

All of our data described here was collected with the visible polarimeter which used a HeNe laser source operating on a wavelength of 633nm. The Mueller matrix described above is found through a relationship between the Fourier coefficients of a series representing the modulated intensity pattern and the elements of the sample matrix. The elements of the Mueller matrix are calculated from a set of equations that use the generated Fourier coefficients. Before beginning the data collection from the polarimeters, I entered a set of Mueller matrix equations into the data reduction program which compensated for inaccuracies in element orientation and nonideal retardances. The same equations had to be entered into the MathCAD simulation programs. These programs helped us to provide data calculated from various errors as an ideal comparison for our experimental data. In MathCAD the operator enters a sample matrix based on the ideal optical element or sample that is being tested. Table 1 contains a list of ideal matrices for various optical elements. The horizontal linear polarizer and the half wave retarder set at forty-five degrees, two of the samples I tested in the visible polarimeter, are among the list of elements in Table 1. Based on this sample matrix and the calibration errors the program calculates the effect of the errors on the ideal Mueller matrix.

Table 1: Ideal Mueller Matrices for Selected Polarization Elements

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

No sample

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Horizontal Linear Polarizer

$$\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Vertical Linear Polarizer

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Half Wave Plate at 45°

RESEARCH DESCRIPTION

Before beginning calibration procedures, the optical elements had to be aligned. The alignment process involved placing a pointed cylinder in a mount on an Ealing optical bench and adjusting the laser to meet the point at any given distance on the bench. By adjusting the front or back of the laser either up or down we could align the laser beam to propagate parallel to the bench. All optical elements were then set to the given height on the optical bench. Positioning the neutral density filter to reflect some of the laser light into the first detector was also an important step included in aligning the laser. After all optical elements were set, I began calibrating the laser by maximizing the laser's power output into the second detector. However, the maximum intensity varied from trial to trial, thus proving to be an inefficient calibration procedure. My next step was to allow the computer to calculate the Mueller matrix on a program designed for sample measurements. *For calibration purposes I ran this program with no sample.* The calibration run would then produce a Mueller matrix that ideally would resemble the identity matrix.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The calibration run would also generate a set of retardance and orientation errors. These azimuthal orientation errors originated with misalignments of the first and second retarders and the second polarizer. Since the error terms for the retarders and the polarizers are a function of

the Mueller matrix and the Fourier coefficients, they can be calculated given one or the other; in this case the identity matrix was the given. Figure 4 represents a typical calibration run. Displayed in this figure are the modulated intensity pattern, a list of Fourier coefficients, a Mueller matrix based on the calculated Fourier coefficients, and a set of orientation errors. Once these error terms were calculated, we could adjust the fast axes of the optical elements to compensate for the errors. We repeated the calibration runs several times to minimize the error terms as much as possible.

Some of my work this summer dealt with rewriting many of the programs used in conjunction with the laser polarimeter. The existing program to calculate the Mueller matrix was designed for the infrared polarimeter. I rewrote the program to be used for the visible polarimeter. This entailed changing the drives that controlled the retarders and polarizers both on the laser setup and in the programs and making it possible to error compensate the Mueller matrix. The computer measurement program was initially designed to calculate the Mueller matrix based on a set of approximate equations, in effect I had to program in a softkey function that allowed the operator to enter the orientation errors from the final calibration run and compensate the Mueller matrix.

After the laser was calibrated, we began taking measurements on samples. The first sample I used was a linear horizontal polarizer. I began by first aligning the polarizer in the sample holder and then maximizing the laser output through the sample. Finally, I ran the Mueller matrix program and calculated a series of Fourier coefficients that in turn gave us the Mueller matrix which I then compensated with the orientation errors from the last calibration run. I repeated this about four or five times and used the measurement run with the most ideal Mueller matrix and the best stability reading. Figure 5 is the data hardcopy representing this measurement run on the first sample. After all of my experimental data was collected from the first sample, I ran a simulation of the polarizer on MathCAD to provide an ideal comparison to my experimental

data. Figure 6 is the simulation of the horizontal polarizer. My second sample was a half wave plate retarder set at forty-five degrees. I followed the same procedure for taking measurements on the retarder as I did for the polarizer. Figure 7 is the data hardcopy of the measurement run on the half wave retarder. I ran a simulation of the half wave retarder, shown in Figure 8, just as I had for the polarizer.

FIGURE 4

VISIBLE POLARIMETER
 12:56:57 31 Jul 1991
 RUN #24 CAL
 ANGLE INCREMENT IS 1 DEGREES
 POWER FLUCTUATION MEASURED ON HgCdTe Photoconductor .3 %
 .001685966155 SUBTRACTED FROM EACH INTENSITY READING.

A 0 = +3.01974451E-01	B 0 = +0.00000000E+00
A 1 = +2.55808332E-03	B 1 = -2.10872803E-04
A 2 = +6.36028517E-02	B 2 = -1.45022566E-03
A 3 = +4.75332132E-04	B 3 = -2.30254303E-04
A 4 = -1.23568645E-01	B 4 = -1.17298532E-03
A 5 = +1.07436789E-03	B 5 = -1.47576911E-05
A 6 = +1.22642977E-01	B 6 = -2.38705558E-03
A 7 = -2.04538720E-03	B 7 = -3.83690382E-06
A 8 = +6.97841439E-02	B 8 = +6.16139939E-04
A 9 = +6.21624182E-04	B 9 = +4.21927910E-05
A10 = +6.20390968E-02	B10 = -7.72737603E-04
A11 = +3.92776880E-04	B11 = +7.06926464E-06
A12 = +5.70406988E-05	B12 = -7.84109818E-05

.973	-0.000	.002	-.001
-.000	1.000	-.005	-0.000
.001	.003	.998	.002
-.001	.002	-.019	.972

RET ORIENT 1	RET ORIENT 2	POL2	RETARD 1	RETARD 2
.305	.209	-.043	93.365	92.649

12:56:57
 31 Jul 1991

VOLTAGE VS. TIME
 NEWPORT STABILITY
 .500 .30%

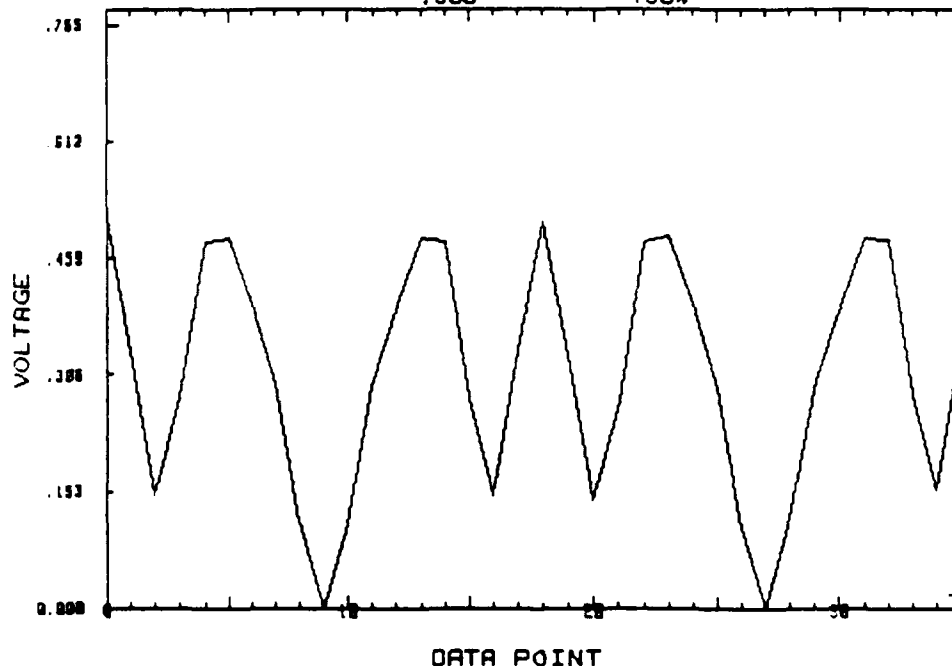


FIGURE 5

VISIBLE POLARIMETER

RUN #5 POLARIZER

13:58:04 31 Jul 1991

ANGLE INCREMENT IS 5 DEGREES

POWER FLUCTUATION MEASURED ON HgCdTe Photoconductor .3 %

.00169 SUBTRACTED FROM EACH INTENSITY READING.

A 0 = +2.66986601E-01	B 0 = +0.00000000E+00
A 1 = +2.75790953E-03	B 1 = +1.63823297E-03
A 2 = +9.91350868E-02	B 2 = -6.93814974E-03
A 3 = -1.25380665E-03	B 3 = +1.64015560E-03
A 4 = -4.54111276E-04	B 4 = -1.30813839E-04
A 5 = +5.10566044E-04	B 5 = +8.97515725E-03
A 6 = -3.52089366E-04	B 6 = +6.70598578E-05
A 7 = -7.96070955E-05	B 7 = +1.53350575E-03
A 8 = +1.75151506E-02	B 8 = +2.72082671E-03
A 9 = +4.23162717E-04	B 9 = -2.27714417E-04
A10 = +9.72524462E-02	B10 = +8.74398804E-03
A11 = +3.96834277E-04	B11 = +3.09653542E-04
A12 = +1.78205191E-02	B12 = +3.54462549E-04

.932	.993	-.050	.009
.984	1.000	-.046	.008
.106	.103	-.008	.001
-.048	-.047	.016	-.006

RET ORIENT 1	RET ORIENT 2	POL2	RETARD 1	RETARD 2
17.475	11.975	-2.464	93.365	92.649

13:59:04
31 Jul 1991

VOLTAGE VS. TIME
NEWPORT STABILITY
.500 .27%

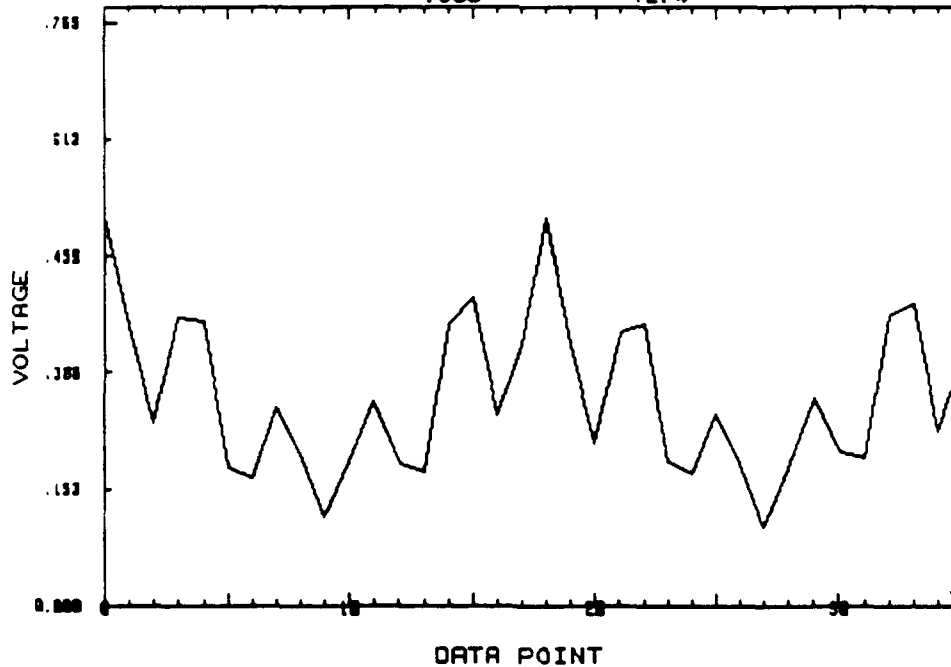
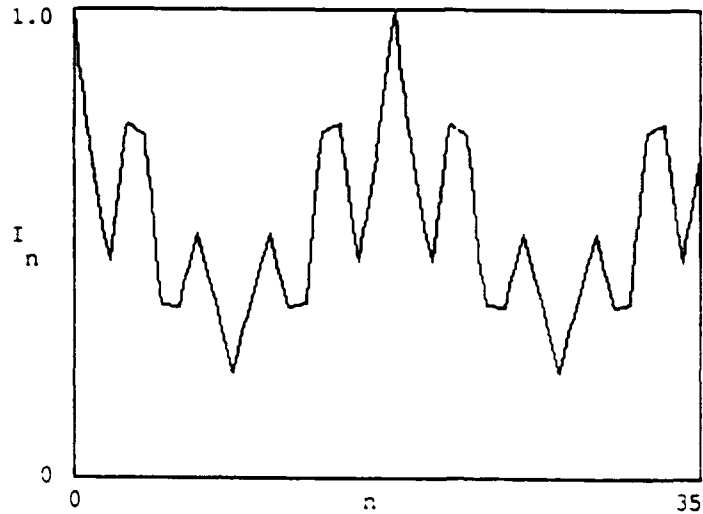


FIGURE 6

$$I_n$$

0.9999746
0.6790617
0.4620056
0.7537841
0.7297123
0.3684342
0.3661384
0.5228142
0.3795931
0.2244615
0.3833091
0.5255413
0.3667013
0.3734596
0.7372335
0.7545688
0.462314
0.6848566
0.9999746
0.6790617
0.4620056
0.7537841
0.7297123
0.3684342
0.3661384
0.5228142
0.3795931
0.2244615
0.3833091
0.5255413
0.3667013
0.3734596
0.7372335
0.7545688
0.462314
0.6848566



$$MM = \begin{bmatrix} 1.0000446 & 0.9999612 & 0.0106465 & 0 \\ 1.0000001 & 0.9999167 & 0.010646 & 0 \\ 0.0072956 & 0.007295 & 0.0000777 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

k	A	B
0	0.542998	0
1	0	0
2	0.1954363	-0.0020808
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0.0346133	0.000064
9	0	0
10	0.1923202	-0.0016918
11	0	0
12	0.0346068	-0.0006729

$$\epsilon_3 = .305 \cdot \text{deg}$$

$$\epsilon_4 = .209 \cdot \text{deg}$$

$$\epsilon_5 = -.043 \cdot \text{deg}$$

$$\xi_1 = 93.365 \cdot \text{deg}$$

$$\xi_2 = 92.649 \cdot \text{deg}$$

FIGURE 7

VISIBLE POLARIMETER

RUN #1 RET

10:47:18 30 Jul 1991

ANGLE INCREMENT IS 5 DEGREES

POWER FLUCTUATION MEASURED ON HgCdTe Photoconductor .4 %

.00192 SUBTRACTED FROM EACH INTENSITY READING.

A 0 = +3.27641642E-01	B 0 = +0.00000000E+00
A 1 = +1.23806489E-03	B 1 = +5.80189426E-04
A 2 = +5.03550025E-02	B 2 = -4.62314241E-02
A 3 = -3.58502407E-03	B 3 = -1.43882239E-03
A 4 = +1.35600173E-01	B 4 = +1.86403058E-02
A 5 = -1.76368290E-03	B 5 = -2.37068360E-03
A 6 = -1.35864604E-01	B 6 = +1.73759939E-02
A 7 = +2.62092538E-03	B 7 = -2.19356758E-03
A 8 = +1.24885854E-03	B 8 = +3.21840161E-05
A 9 = -1.58544447E-03	B 9 = -1.12571103E-03
A10 = +6.17322901E-02	B10 = -3.21050512E-02
A11 = +4.50530442E-03	B11 = -6.94809605E-04
A12 = +5.84465459E-02	B12 = -5.25812220E-02

1.000	-.031	.017	.001
.058	.799	-.678	.003
.093	-.678	-.766	-.043
-.003	.024	.043	-.989

E3	E4	E5	D1	D2
7.238	-10.395	-6.801	93.605	92.306

10:47:18
30 Jul 1991

VOLTAGE VS. TIME
NEWPORT STABILITY
.540 .36%

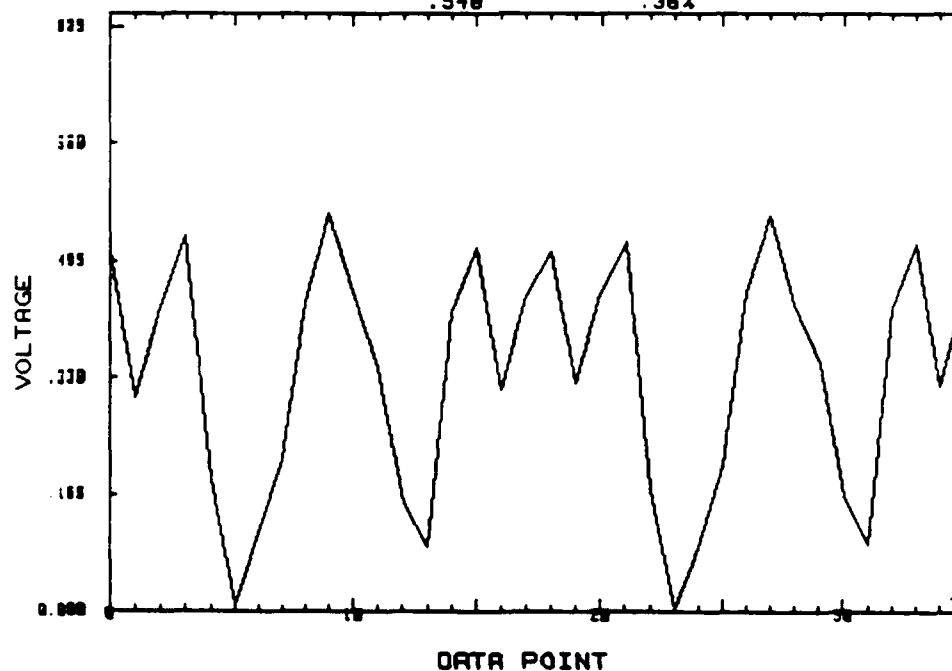
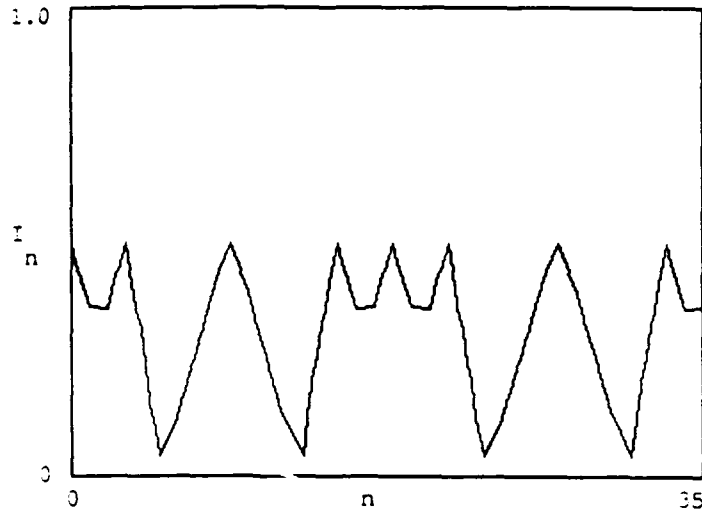


FIGURE 8



$$MM = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.9999985 & 0.0017453 & 0 \\ -0.0008727 & 0.0017453 & -0.9999985 & 0 \\ 0 & 0 & 0 & -0.9999996 \end{bmatrix}$$

A		B	
k	k	k	k
0	0.3125	0	0
1	0	0	0
2	0.0624999	-0.0001091	-0.0001091
3	0	0	0
4	0.125	0.0001091	0.0001091
5	0	0	0
6	-0.125	0.0001091	0.0001091
7	0	0	0
8	0	0	0
9	0	0	0
10	0.0625	0	0
11	0	0	0
12	0.0624999	-0.0001091	-0.0001091

$\delta 1 \approx 90^\circ$

$$\delta 2 \approx 90 \cdot \text{deg}$$
$$\delta 2 \approx 90 \cdot \text{deg}$$

RESULTS/DISCUSSION

The results I obtained from the visible polarimeter were conclusive in that they proved to be similar to the ideal comparisons from MathCAD. Both the polarizer and the retarder helped in demonstrating the effectiveness of the visible polarimeter and in defining the accuracy of the measurements taken on the polarimeter. As far as the experimental and ideal comparisons went, the Mueller matrix from the horizontal polarizer resembled the MathCAD ideal matrix much more closely than the half wave plate.

$$\begin{pmatrix} .932 & .993 & -.050 & .009 \\ .984 & 1.000 & -.046 & .008 \\ .106 & .103 & -.008 & .001 \\ -.048 & -.047 & .016 & -.006 \end{pmatrix}$$

Polarizer(experimental matrix)

$$\begin{pmatrix} 1.0000446 & .9999612 & .0106465 & 0 \\ 1.0000001 & .9999167 & .010646 & 0 \\ .0072956 & .007295 & .0000777 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Polarizer(ideal matrix)

$$\begin{pmatrix} 1.000 & -.031 & .017 & .001 \\ .058 & .799 & -.678 & .003 \\ .093 & -.678 & -.766 & -.043 \\ -.003 & .024 & .043 & -.989 \end{pmatrix}$$

Retarder(experimental matrix)

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & .9999985 & .0017453 & 0 \\ -.0008727 & .0017453 & -.9999985 & 0 \\ 0 & 0 & 0 & -.9999996 \end{pmatrix}$$

Retarder(ideal matrix)

The matrices above show the differences in the MathCAD and experimental values obtained. In the first set of matrices for the polarizer, the four corner elements of the experimental matrix should ideally be one. The elements are off by an average factor of four percent. In the error compensated MathCAD matrix the element error is less than one part in ten thousand. In the experimental matrix for the retarder however, the average percent error of the elements is about seventeen percent while the MathCAD simulation is less than one part in one hundred thousand. In the actual lab setup due to the fact that the sample mount was a manually controlled rotation device, the alignment of the samples was far from precise which affected the experimental matrices.

CONCLUSION

Through the duration of the summer, I built and operated, calibrated and took measurements on, as well as designed and rewrote programs to coordinate with a HeNe visible laser polarimeter. My project is an adaptation of the research in infrared laser polarimetry which has been an ongoing research project at Wright laboratory. Before I began building the visible polarimeter, I had to gain a basic understanding of polarimetry. I accomplished this by beginning my summer work in assisting with the CO₂ infrared laser polarimeter. I helped calibrate and take some of the preliminary measurements on it. Basically as a summary of my summer tasks, I built the visible polarimeter. This included aligning it, calibrating it, and taking measurements on two samples: a horizontal linear polarizer and a half wave plate retarder. Some of my other tasks were modeling the polarimeter with computer simulations and rewriting existing programs to work with the visible polarimeter.

Several important uses for polarimetry exist, however, one of the most important being the need to characterize electrooptic modulator materials. This includes materials for processing systems using polarized and infrared sources, target simulation, pattern recognition systems, and optical computing. Laser polarimetry may also assist in the understanding of certain crystal data: crystal structure, plane orientation etc. In effect, the field of laser polarimetry offers a variety of applications ranging from small scale testing resembling my summer project to the testing of complex materials and samples.

In conclusion, my work this summer was productive in that the visible polarimeter has been built and records of its calibration and first measurements were kept so that future research in the field of visible laser polarimetry may be enhanced and more complex samples may be tested in the polarimeter for possible publication.

In conclusion, my work this summer was productive in that the visible polarimeter has been built and records of its calibration and first measurements were kept so that future research in the field of visible laser polarimetry may be enhanced and more complex samples may be tested in the polarimeter for possible publication.

SUMMER EXPERIENCES/LESSONS LEARNED

During this summer I have accumulated so much knowledge and understanding of fields that require years of college to grasp. The High School Apprenticeship Program has allowed me to experience firsthand the real world in respect to everything I have ever learned. Day to day I faced challenges that required a broad spectrum of learning and application of what I'd learned. To have the opportunity to be able to not only see but experience in full what life as an engineer might be like, is a great asset of the High School Apprenticeship Program to which I'm grateful for. This summer I learned the fundamental concepts of optics from the definition of a light wave to the mathematical representation of light. I gained a vast array of computer programming skills in addition to learning how to efficiently use several software programs. My summer experiences took me deep into the field of laser polarimetry and allowed me to gain a complete understanding of its purpose and formulation. I obtained valuable experience in operating the lasers and running the polarimeters. The lessons I learned this summer involved not only theoretical and experimental techniques in optics but also diligence, persistence, patience, and respect. I hope that next summer I will be able to complete more work on the polarimeters based on the lessons I have learned and the experience I have gained this summer in the High School Apprenticeship Program.

REFERENCES

1. Chenault, David B., "Mueller Matrix Infrared Polarimetry", Final Report for the 1988 USAF-UES Graduate Student Research Program, 22 August 1988.
2. Hodgson, Randall R., "Laser Polarimeter Development", Final Report for the 1989 USAF-UES Graduate Student Research Program, 17 August 1989.
3. Gove, Randy, "Infrared Laser Polarimetry", Final Report for the 1990 USAF-UES Graduate Student Research Program, 22 August 1990.
4. Goldstein, Dennis H., "Polarization Modulation in Infrared Electrooptical Materials", dissertation proposal submitted to Department of Physics, University of Alabama in Huntsville, 1989.

MEASUREMENTS TO DETERMINE MECHANICAL CHARACTERISTICS OF MATERIALS

High School Apprentice: Jason A. Kitchen

My summer work assignment was in the Warheads Branch of the Armament Directorate at Eglin Air Force Base. The work involved the use of physical measurement techniques to determine mechanical characteristics of various materials. This research included microhardness testing of impact specimens and high-speed film data analysis. Both tasks were related to material science, a field in which my interest has increased as a result of this summer's work. This research answered some original questions and raised new questions to be examined during future research.

I would like to express sincere thanks to Joel House, my mentor for this program. He did his best to introduce me into the current work of the Warheads Branch and allowed me to become a contributing member of his research team. Thaddeus Wallace, the laboratory technician in the area where I did much of my work, was also a great help throughout the summer. Steve Hatfield and Jessica Mayes, graduate students from the University of Kentucky, shared their knowledge of materials science for my benefit, as well as being exciting companions for the research program. I would also like to thank all the people from the Warheads Branch of the Armament Directorate for helping me and assisting me in every way they could throughout my stay. Finally, I would like to thank and congratulate Don Harrison for his tremendous job in leading the High School Apprenticeship Program. Mr. Harrison proved to be a great coordinator and help to everyone involved in the program.

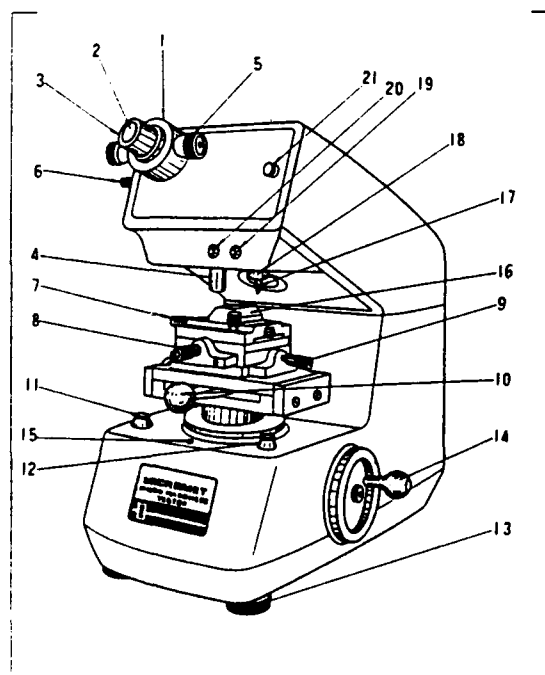
The primary objective of the Warheads Branch is to use new technologies and materials to develop more effective warheads. In order to accomplish this objective, research is conducted in the realm of physical and mechanical characteristics of materials being considered for use in warheads. There are a variety of hydrocodes (computer simulation models) used to simulate the interaction of different materials, geometries, and warhead processes. These hydrocodes can be used to develop future warheads, as well as to improve current ones. Before these computer models can be useful, however, they must accurately predict all of the variables which take place when a warhead penetrator interacts with a target. To assess the validity of current hydrocode models, experiments are done to determine material characteristics.

One experiment that simulates the high rate deformation process which occurs in warheads is the Taylor Anvil test. In this procedure, a plane-ended cylindrical rod of a material is shot from a .30 caliber rifle into a steel target called an anvil. This rod, usually around 1.0 inch long and .297 inch diameter, is of a material which is being researched. After the rod impacts with the anvil, it can be analyzed to determine its physical and mechanical characteristics. This data can then be compared to the predictions of hydrocode models to assess the accuracy of the prediction against true experimental evidence. A high-speed camera with a maximum speed of two million frames per second is aimed at the area where the Taylor rod impacts the steel anvil. Eighty-two frames of film are exposed, showing the trajectory and impact of the rod. There are two pressure transducers mounted in the barrel of the rifle. As the rod goes down its path in the gun, a pressure reading is taken from each transducer as the specimen passes. In the same way, two laser beam sensors are positioned in the impact area. Laser beams are successively cut by the rod as it travels in their path. Both the pressure and the laser sensors serve as velocity measuring devices.

The first of my summer research tasks was microhardness testing on a group of Taylor Anvil impactors. The objective of these tests was to compare the varying hardness regions of a sample with data from hydrocodes. There are two main components in a microhardness machine: a diamond indenter and a microscope (see Figure 1). The diamond indenter can be placed under a variable load, causing the diamond tip to press down and make an indentation in the test specimen. This indentation is then measured using the microscope. The two diagonals of the square indentation are measured in microns and then averaged. This average and the load in grams are then entered into an equation which gives a Vickers Diamond Pyramid Hardness value. This process is then repeated at regular intervals across the specimen to measure variances in hardness, caused by plastic deformation at the anvil face. The hardest regions occur closest to the target-specimen interface, due to this deformation.

Figure 1

MICROMET Microhardness Tester

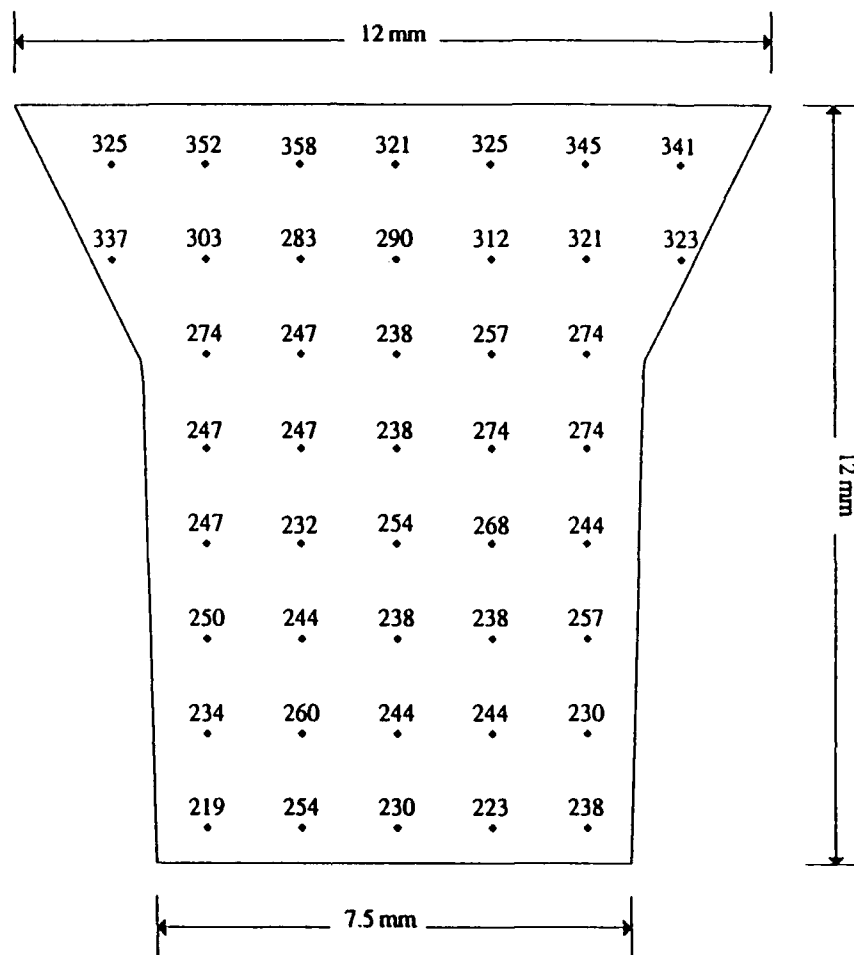


1. Measuring Microscope
2. Measuring Eyepiece (15x)
3. Scale Adjusting Knob
4. Objectives (40x or 6x)
5. Micrometer Measuring Knob
6. Viewing Light Intensity Adjustment Knob
7. Vise Tightening knob
8. Micrometer Screw for Carriage (Front and Rear)
9. Micrometer Screw for Carriage (Right and Left)
10. Carriage Shifting Lever
11. Main Switch
12. Switch for Automatic Load Application
13. Leveling Feet (3)
14. Handle for Elevating Stage
15. Level
16. Vise
17. Diamond Indenter
18. Indenter Holder
19. Green Indication Bulb
20. Red Indication Bulb
21. Weight Chamber Cover

My second task of the summer was the analysis of high-speed film data taken from several Taylor Anvil test shots. For many tests, the velocity readings from the pressure and the laser sensors did not correspond, so a better means of determining velocities was needed. By analyzing film data from past Taylor Anvil tests, another velocity reading could be calculated to assess whether the pressure sensors or the laser sensors were more accurate. Each film frame showed the rod as it got closer to the anvil target. With the time intervals between frames known, the displacement between the rod and anvil was measured for each frame. When the displacement versus time values were plotted and a linear regression was computed, the velocity of the impactor could be determined.

Several conclusions were drawn from the microhardness testing. There proved to be a relationship between varying hardness regions in the Taylor impactors and the mechanical characteristics plotted from hydrocodes. The widest ranges were found in the steel Taylor samples where thermal effects played a smaller role than in the copper and aluminum samples (see Figure 2). Thermal effects refer to the volume of heat generated when metals are plastically worked. Estimates have been made that 95% of the energy to deform material is observed as heat energy, the balance being stored as internal or strain energy. This strain energy is reflected in the hardness values of the metal. Thermal conductivity had a greater affect on the copper and aluminum samples, causing smaller hardness ranges. The data obtained from the microhardness tests proved the utility of using microhardness regions to examine possible relationships between stress, strain, and other mechanical properties of materials. Future research into these relationships may prove that microhardness testing is an accurate way of validating hydrocode predictions.

Figure 2
UK-191 Steel Taylor Impact Specimen



Results of film data analysis were ambiguous. The velocities calculated from film data differed from both the pressure and the laser sensor readings (see Figure 3). This failed to prove whether the pressure or the laser sensors were better at determining the velocities from the Taylor Anvil tests. Film data analysis may prove to be a better means for velocity determination in future Taylor Anvil tests.

Figure 3

Taylor Anvil Velocity Measurements

Test No.	Pressure	Laser	Film
JG-46	786 f/s	---	690
JG-49	744 f/s	730 f/s	716 f/s
JG-50	675 f/s	656 f/s	640 f/s
JG-51	527 f/s	539 f/s	502 f/s

During the course of the summer I acquired many beneficial skills and knowledge. Through the work I performed I gained fundamental knowledge regarding the basic concepts of materials science. I formatted and ran my own analysis during several research projects. I was introduced to a variety of tests and laboratory devices used to determine the physical and mechanical characteristics of materials. I utilized a variety of IBM software, including Gem Presentation Software, LOTUS 123, and Grapher. This broadened my knowledge in the field of scientific-related computer programs as well as the format of IBM machines. I operated several laboratory analytic devices such as a microhardness tester, an optical comparator, and several microscopes. I interpreted data from these devices and learned how they relate to the characteristics of a material. This summer's research increased my knowledge of math and science. It taught me basic skills in organizing and presenting a seminar briefing, and honed my skills in writing technical reports. Overall, this job gave me the opportunity to experience a career as a research engineer, which to me is the most valuable asset to be gained from participating in the High School Apprenticeship Program.

BIBLIOGRAPHY

Buehler, Adolph I. Instructions for Micromet Micorhardness Tester. Buehler LTD,
Evanston, IL. 1977.

Carrington, W. E. and Mary L. V. Gayler, D. Sc. "Changes in microstructure caused
by deformation under impact at high-striking velocities." The use of flat-
ended projectiles for determining dynamic yield stress (sic). 24 July 1947.

NEURAL NETWORK IMPLEMENTATION OF AN EXTENDED KALMAN
FILTER

HIGH SCHOOL APPRENTICE:DARAN MASON

MENTOR:CRAIG EWING

WL/MNSI

8-9-91

Abstract

Neural network simulation software was used to apply neural networks to the problem of an Extended Kalman Filter. Three different networks were used in training. The first was a direct replica of an Extended Kalman Filter, five inputs and nine outputs with a hidden layer of nine added. This network failed to learn thoroughly. In the second network a sliding window was added to take the time dependent nature of the problem into account. This changed the input layer to twenty-five with two hidden layers of twenty-five and an output layer of nine. Although this network learned, when tested it failed. The last network divided the nine nodes in the output layer into three groups of three. One group was associated with the parameters for each direction (x,y,z). Each one of these networks had an input layer of two, output layer of three, and a hidden layer of eight. These networks failed to learn. Future work should involve experimentation with the back-propagation network or a new type of network such as a time dependent network.

NEURAL NETWORK IMPLEMENTATION OF AN EXTENDED KALMAN FILTER

High School Apprentice : Daran Mason

Neural networks consist of neurons or nodes which are based on neurons in the brain. These nodes are put into layers which are connected to other nodes in different layers. Each node processes its inputs to produce an output which is carried through connections to other nodes. As in biological neurons the strength of the connections can change. The way nodes are connected, the function each neuron implements, and how strong each connection is determines how the neural network will learn. Neural networks are used in a wide range of applications from scoring loan applications to brain of the terminator (Arnold Schwarzenegger) in the movie Terminator 2. There are over one hundred different types of neural networks and newer ones coming every day.

A back-propagation neural network was used to implement an Extended Kalman Filter (EKF). The EKF estimates relative position, relative velocity, and target acceleration for the guidance law of an interceptor. The network was trained by a Perfect Kalman Filter which provides truth data.

The Kalman Filter is an estimator for the guidance law of an interceptor. It has five inputs and nine outputs. The inputs are azimuth, elevation, x-acceleration, y-acceleration, and z-acceleration of the interceptor. The outputs are x relative position, y relative position, z relative position, x velocity relative, y velocity relative, z velocity relative, x acceleration target, y acceleration target, and z acceleration target.

I shall explain neural networks by using an example network. We will be using a back-propagation network. There are so many different types of networks it would be hard to try to explain them all. Let's say your trying to distinguish between three different types of iris flowers. The data you have is petal length,

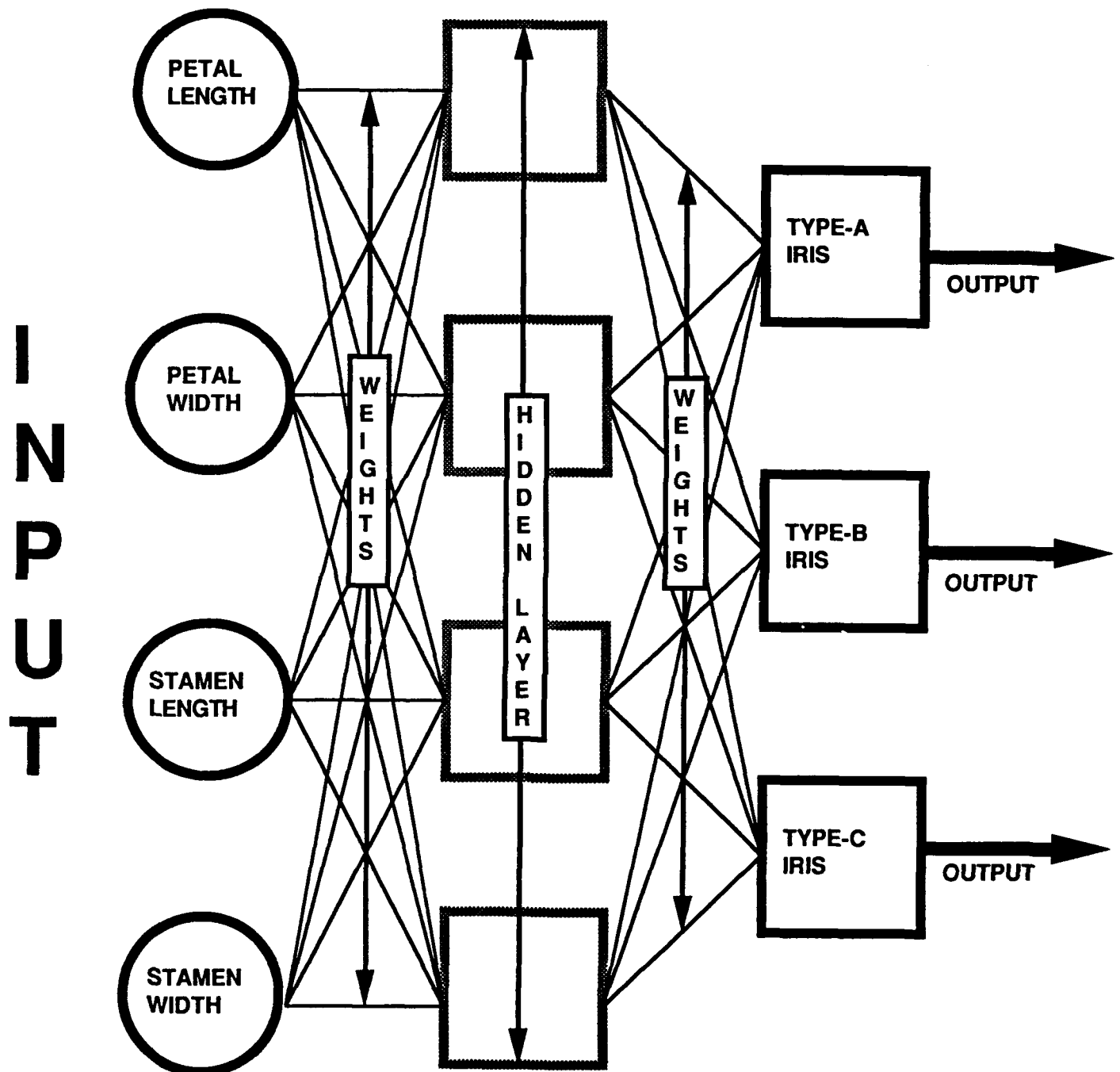
petal width, stamen length ,and stamen width for each type of iris. The lengths and widths would be the inputs and the output would be the three different types of flowers. A hidden layer of four nodes will be added to help the network learn.(see figure 1)

There are two steps in neural networks, learning (or training) and testing. In training the network is given the inputs and the outputs the network should get. In the figure you can see the nodes are connected to each other. In these connections are small variables called weights. Each input is multiplied by its connected weight and then added to the other multiplied weights at a node in the hidden layer. This number is then passed through a nonlinear transfer function. Then the output is multiplied by its connected weights and the process is repeated at the output layer. The network will keep changing its weights until the correct output is obtained. Once the network has learned, it will be tested with a different set of data. The networks outputs are compared with the outputs it should have gotten to see if the network has learned.

In the Kalman Filter network experiment NeuralWorks professional II/plus software was used on a Macintosh IIX computer. Data was downloaded from the Perfect Kalman Filter program on a Vax computer for training and testing. Three different networks were tried. The three networks evolved as a result of the trial and error process of designing neural networks.

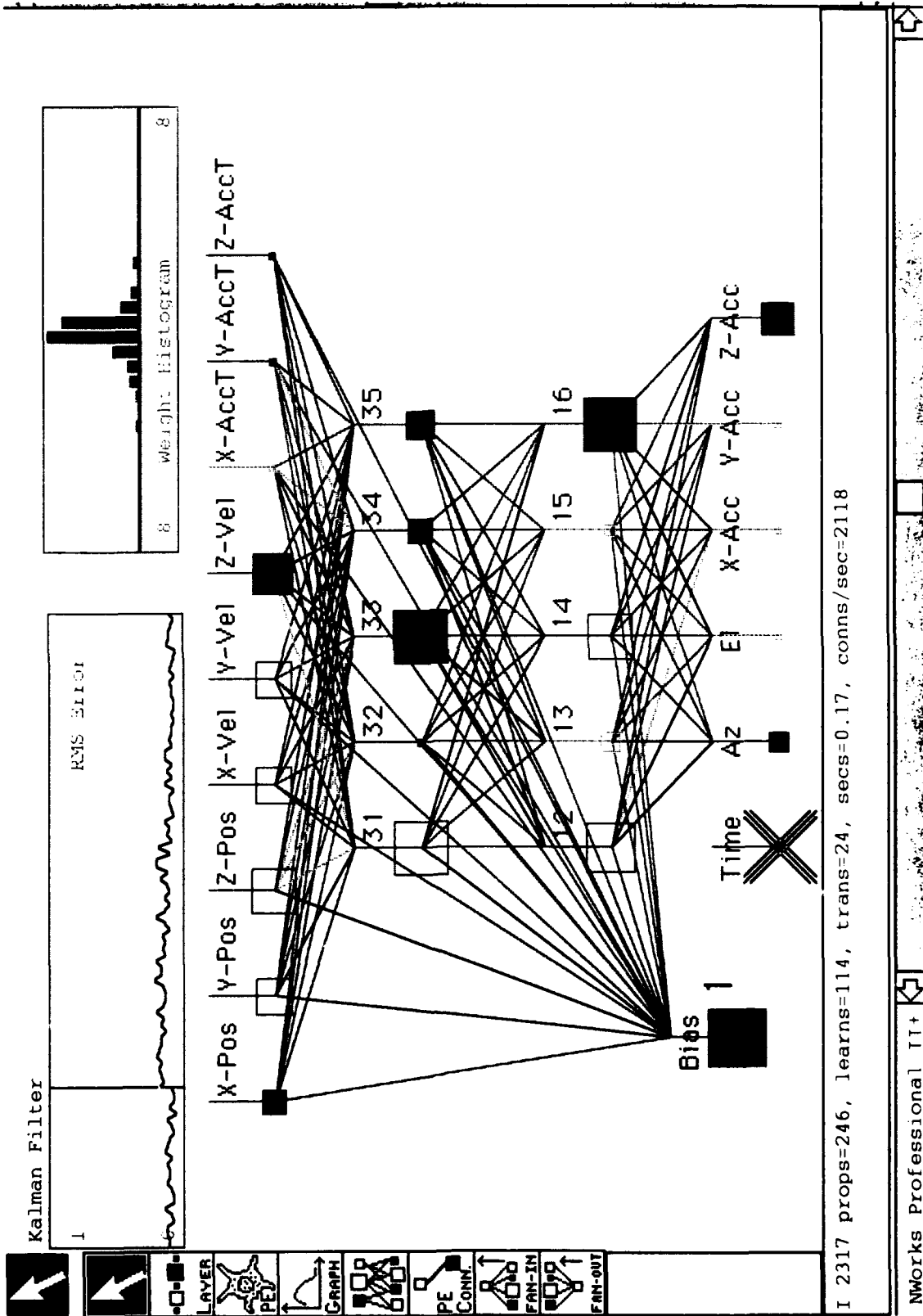
The first network was a direct implementation of the Extended Kalman Filter .(see figure 2) The only exception was that a hidden later was added. In the Perfect Kalman Filter program there are four different aspect angles for the interceptor. These angles are 0 degrees, 45 degrees, 90 degrees, and 135 degrees where 0 degrees is a head on intercept course. In the first network one or more data files from each aspect angle were appended together and used for training.

In the second network a sliding window was added to the input layer. The network then used inputs from the last five time steps. This changed the input layer to twenty-five, five sets of five inputs. At each new time step, a new set of inputs is brought in and one oldest set of five is taken out. This helps the network



(figure 1)

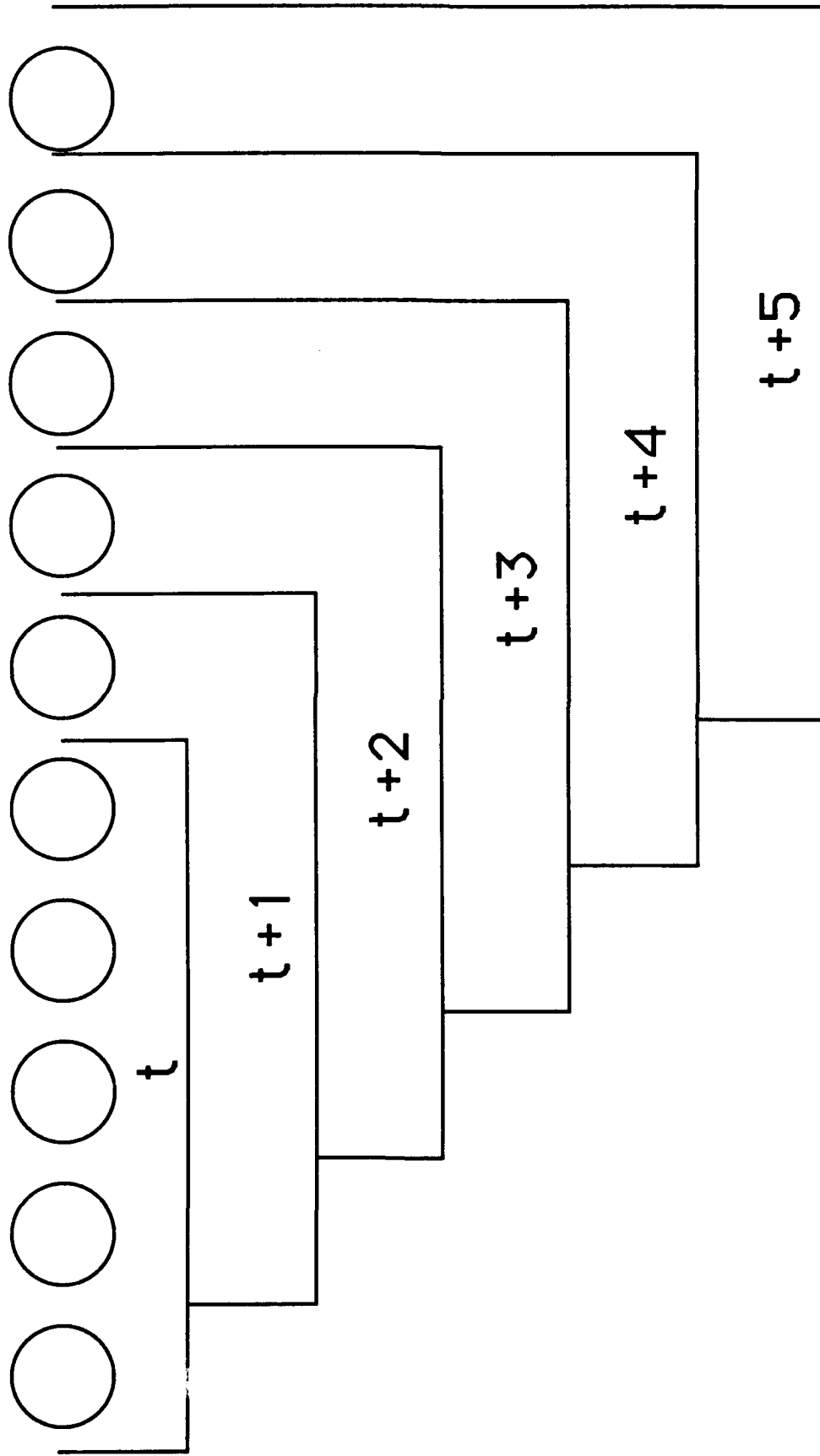
NEURAL NETWORK



learn, because it has data involving the time dependent nature of the filter.(see figure 3) The final setup for this network was a input layer of twenty-five, two hidden layers of twenty-five, and an output layer of nine.(see figure 4)

In the third network the Xs' , Ys' , and Zs' were divided in an attempt to isolate the problem in each direction. The Kalman Filter has more trouble from the Xs' than the Ys' and Zs' . This might be the same problem for the neural network. The input layer was reduced to two. For example, elevation and x accceleration were the inputs for the X group and the outputs were x relative position, x velocity relative, and x acceleration target. This experiment also had one hidden layer of eight.(see figure 5)

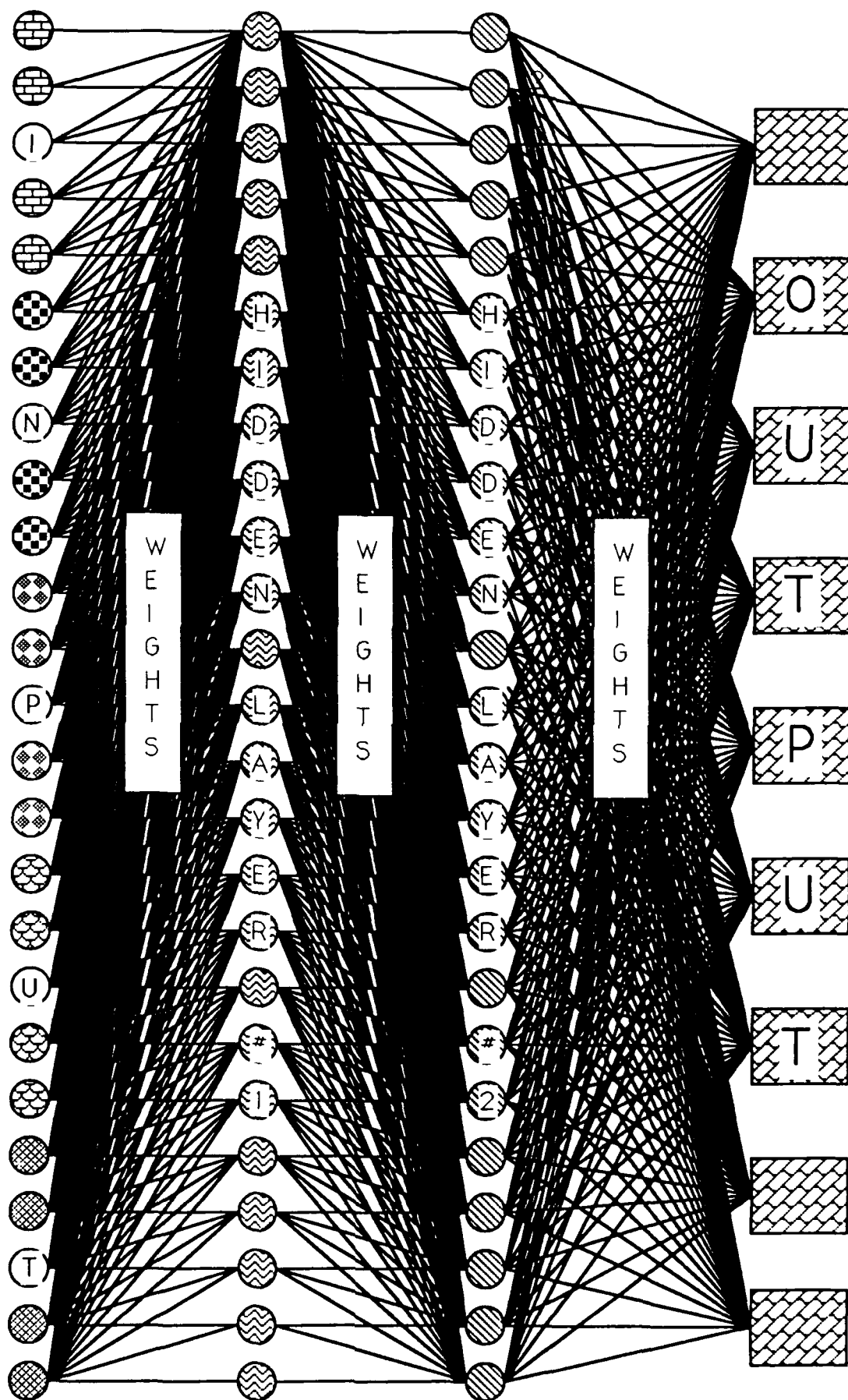
SLIDING WINDOW



O = 5 inputs

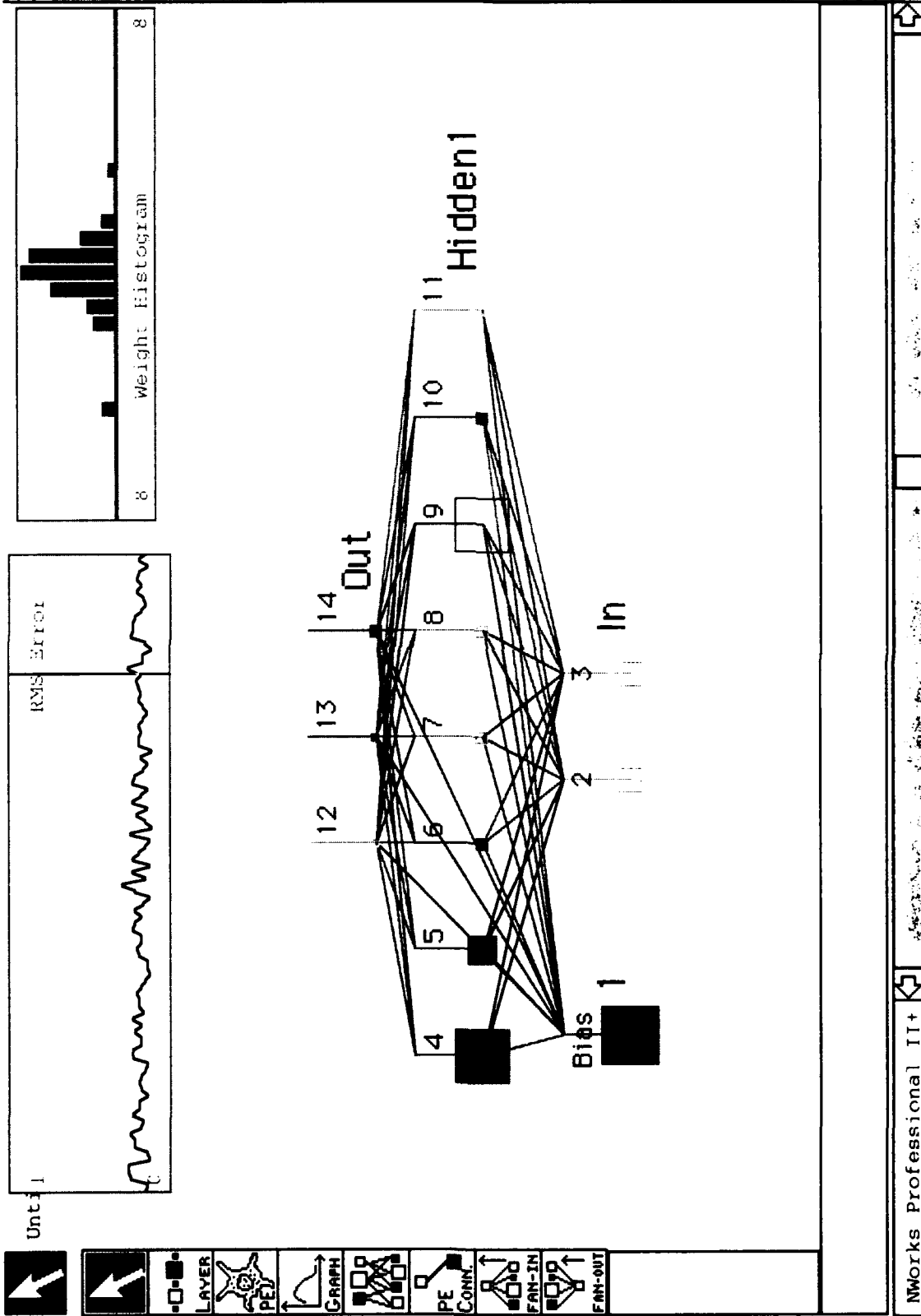
K
A
L
M
A
N

F
I
L
T
E
R



N
E
U
R
A
L

N
E
T
W
O
R
K



Results

In the first network the network learned, but not thoroughly. When tested, the network failed. When comparing the actual results to the desired results. A lot of the error appeared in x acceleration target. The error was small as six percent and as large as four hundred thirty-one percent.(see figure 6) In the figure there are three columns. The first shows the desired output, the second is the actual output from the network, and the third is the error between the two. The second network also learned very well, but failed In testing.(see figure 6) In the graph y is error in meters and x is error over time.The graph shows network output (A) and the desired output from the PKF (D) for the output x velocity relative.The third network failed to learn totally. (see figure 5) In the up-left-hand corner of the figure the is a graph titled RMS Graph. This graph shows the overall error of the output layer. As you can see the graph never reached zero, so the network didn't learn.

	Desired	Actual	Error
1	97378.094	57377.781	-41.077
2	-1676.063	208.582	-112.445
3	-855.125	360.318	-142.136
4	-10087.808	-9389.342	-6.924
5	-381.361	-190.514	-50.044
6	267.471	-17.200	-106.431
7	13.047	69.322	431.325
8	76.043	84.464	11.074
9	-44.545	-40.439	-9.218
10	96873.711	57490.348	-40.654
11	-1694.913	209.110	-112.338
12	-841.931	360.743	-142.847
13	-10087.264	-9389.521	-6.917
14	-372.619	-189.950	-49.023
15	260.273	-17.094	-106.568
16	13.068	69.227	429.744
17	76.166	84.547	11.004
18	-44.617	-40.513	-9.198
19	96369.367	57569.105	-40.262
20	-1713.325	209.452	-112.225
21	-829.098	361.058	-143.548
22	-10086.652	-9389.722	-6.909
23	-363.849	-189.575	-47.897
24	253.061	-17.018	-106.725
25	13.089	69.043	427.489
26	76.289	84.601	10.895
27	-44.689	-40.577	-9.201

NEURAL NET EKF IMPLEMENTATION STUDY

$\times 10^5$

1.25

D 1 DESIR

A 1 ACTUAL

1.00

0.75

0.50

0.25

0.00

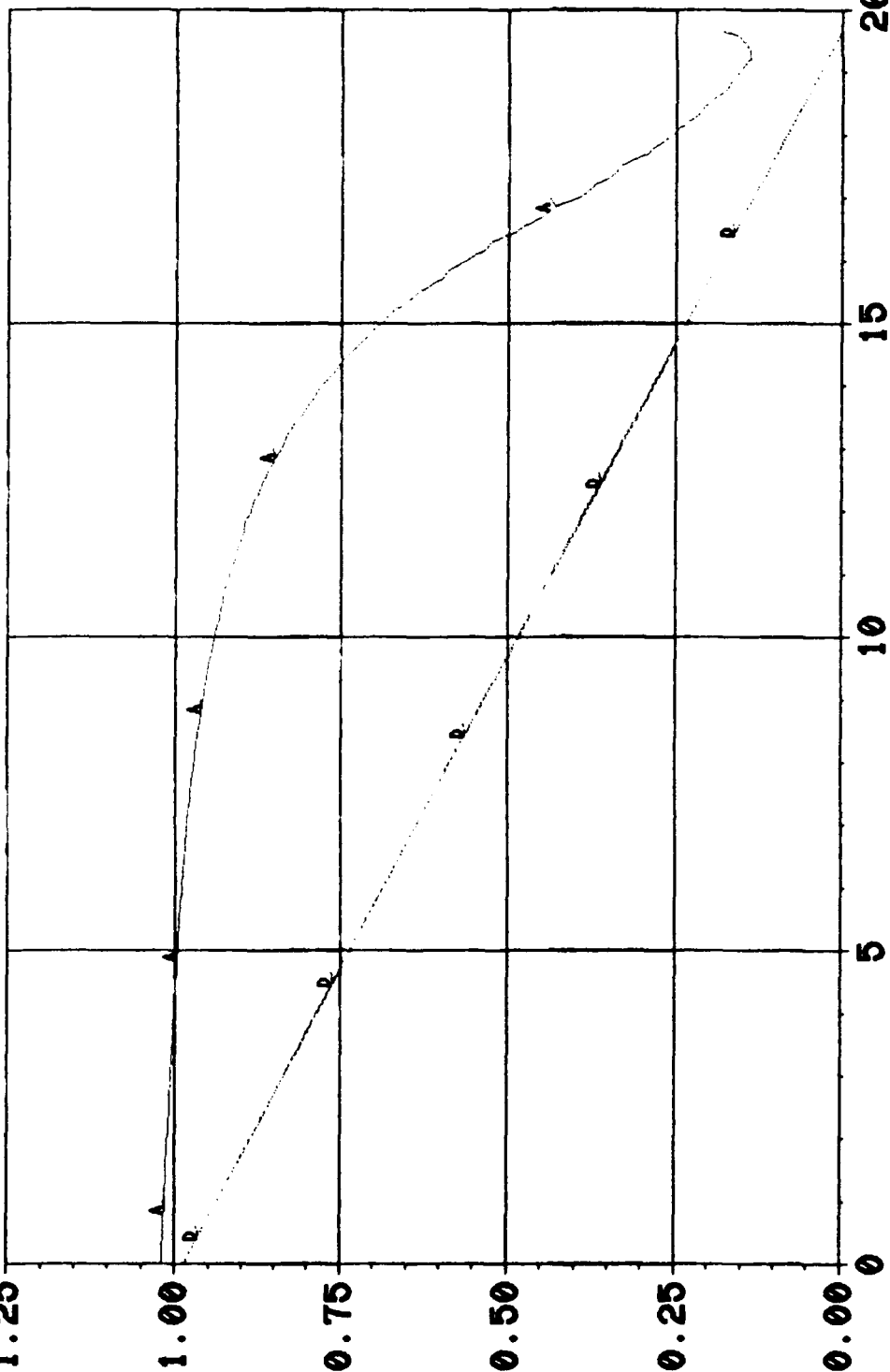
0

5

10

15

20



D 1 TIME A 1 TIME

CREATED BY:
 NAME: Hoon
 ORG : NNSI
 DATE: 9-AUG-01
 TIME: 18:58:48

FILE 1 =

Conclusion

It is unclear why the networks failed to properly implement the filter. More experimentation with a back-propagation network or another type of network, such as a time dependent network, is necessary to come up with the proper structure. I think it is possible for a neural network to implement a Kalman Filter and the proper network structure could be found with further experimentation.

Acknowledgements

I would like to thank the following people:

Craig Ewing

1 Lt. Kurt Bolin

Matthew Willis

Anne Carstens

QUICK-LOOK, VIDEO-BASED SCORING FOR CAPTIVE FLIGHT TESTING OF ASARG

CHRISTINE D. RIENDEAU
JOHN WOLVERTON, MENTOR
ASARG PROGRAM MANANGER
16 AUGUST 1991

I. Introduction

The summer of 1991 was my second summer in the high school apprenticeship program at Eglin Air Force Base. During this summer, captive flight tests were conducted of the Autonomous Synthetic Aperture Radar Guidance (ASARG) program, and I was privileged to be part of the team. My project this summer was to develop image processing techniques to accomplish quick-look analyses of the data collected during the missions, and I was also asked to help with much of the set up for the captive flight tests. I worked in WL/MNGA, advanced guidance division, advanced development section, under the mentorship of John Wolverton and the guidance of Captain Jeffery L. Owen and Air Force Academy Cadet Anne Clark.

II. Acknowledgments

Mr. John Wolverton, mentor and ASARG program manager

Capt. Jeffery L. Owen

A.F. Academy Cadet Anne Clark

H.S.A.P. program director Don Harrison and also Dr. Klausutis for the privilege of being part of the program

WL/MNGA Advanced Development Section: Al DiSalvio, Bill Eardley, Capt. Mitch Hicks, Jimmy Carter, Tim Jones, Capt. Pat Gardner, Capt. Tom Connare, Mr. Rich Porter, 1Lt. Maggie Fenton.

WL/MNGA TEAS support: Greg Christakos, Donny Cates

Lesha Denega, WL/MNMF high school apprentice

III. Background

The Autonomous Synthetic Aperture Radar Guidance program is the development of an advanced guidance system for a weapon that is launched beyond visual range of the target. It automatically guides the missile to the target and allows the plane launch-and-leave capability. The ASARG seeker is guided by its continually updated inertial navigation system (INS), which uses a template reference to calculate the position of the target. ASARG is in its third phase of development. During phase I, the SAR seeker and its components were designed and fabricated, and then tested in the rooftop facilities of each of the two government contracting companies involved in the ASARG program. The synthetic aperture radar (SAR) offers several advantages over real beam radars and infrared sensors. SAR radars collect their data over a longer flight path, and after range-gating and phase detecting the collected data, the doppler history of the sequence is processed to form the SAR image. With this process, a SAR may obtain an image with resolution attainable only in real beam radars that have such a large antenna aperture (and therefore a small beamwidth) that they do not conceivably fit in the cone of a missile. Another advantage of the ASARG system is that it is an all-weather system, provided sufficiently long wavelengths of energy are chosen.

Phase II of ASARG included integrating the laboratory tested

system into pods beneath an aircraft designed for captive flight testing of the radar. Captive flight tests for phase II were conducted at the two contractor locations: the Loral corporation in Phoenix, Arizona, and the Raytheon company in Boston, Massachusetts. At the completion of phase II of ASARG, most major software and hardware problems had been worked out, although Loral's seeker continued to have problems with the Carousel IV INS system.

ASARG is currently in phase III of its development. The third phase of the program consists of captive flight testing of the SAR against high value ground targets located near Eglin Air Force Base. The tests are split into two six weeks periods. The first six weeks allows the companies to locate and solve any problems with the system. During the second period, primary targets that have not yet been flown against are tested, and the results are scored. These results are used to report the success or failure of the ASARG system, its accuracy, and its possible future.

The objectives of the ASARG captive flight tests fall within the area of terminal guidance. Two objectives are to evaluate autonomous target acquisition and aimpoint selection. The final objective is to compare the three methods of scoring accuracy: video-based, SAR-based, and mathematical methodologies. The mathematical scoring is the most accurate and is vector-based. This task will be accomplished by the 3246TESTW/SC for each mission. The SAR-based scoring method will be accomplished by

WL/MNGA TEAS support using digital SAR imagery and processor data.

My project for the 1991 summer was to develop the video-based aimpoint selection/scoring method for the third phase of ASARG captive flight testing.

IV. Description of Research

The video-based scoring method was developed with the knowledge that two types of data could be obtained: digitized SAR imagery from the seeker and VHS magnetic tapes from the TV camera attached to the plane beside the SAR seeker and slaved to the boresight of the seeker. The purpose of the video-based scoring is to quantify the boresight error to support the government quick-look analyses. A final objective of this research was to establish the merit of the video-based aimpoint selection process relative to other scoring methods.

The first step in the development of the process was to obtain an enlarged SAR image of a single track. The data file for each run of a flight contained small images of each update (totaling twelve tracks), a final SAR image, and the graph of the correlation between the template and the actual image (see appendix 1). Software, written in FORTRAN, that had been previously written and supplied by WL/MNGA TEAS support was modified to capture the single track, enlarge it to fit the screen of a GOULD IP8500 monitor, and center it on the screen

(see appendix 2). Before a transparency/hardcopy was made of this image, color processing was applied to remove the blue background color from the image, leaving a clear background so that the image might be overlaid on a hardcopy of the video image to compare the two. However, the SAR image is on a range-doppler coordinate system, and the video image is azimuth-elevation (or angle-angle). For the comparison of overlaid images to be correct, the images would have to be transformed to the same coordinate system. Finally, the software that was used to develop this method was written for the phase II captive flight test imagery. Extensive modifications had to be made in the SAVE_MAP subroutine for the phase III data files, and so the subroutine within the RED_DATA file was commented out and a new file with a modified SAVE_MAP subroutine was created and linked with a new SAR_IMG1 file that enlarged the track by seven times (see appendix 3).

The next part of the method was to use the digitizing equipment and software in the image processing laboratory of the armament directorate to capture a video frame. A procedure was discovered to highlight the crosshair, aimpoint selection box, and outlines of targets in video frames to make those images more easily distinguishable in the hardcopy image.

Part of the project was to develop a user instruction manual so that a user with no VAX/VMS or SAR experience could duplicate the process. This user manual will be modified for phase III software, and includes a list of equipment and comments on the

subroutines (see appendix 4).

V. Discussion

The first step in the development of the video-based aimpoint selection methodology has been completed. The next step will be to read the SAR and video data files to determine the pixel that is considered the aimpoint in each image and to mark it in the digitized image. Phase III SAR images have the aimpoint marked with a small crosshair, but it is lost during the capture of a single track in the SAR_IMG program. Finally, the SAR image and digitized video frame must be transformed to a common coordinate system so that they may be geometrically aligned using the pixel designated as the aimpoint, a common point that they share.

Two problems have hindered completing this process. The program containing some of the data necessary to the transformation of the range-doppler SAR image and the azimuth-elevation video frame has been obtained, but more such data is needed. Also, the transformation is not possible for missions where the aimpoint is offset greater than about 800 feet from the trackpoint, because this puts the aimpoint out of range of the SAR image.

VI. Conclusion

With the help of WL/MNGA TEAS support the video-based scoring method should soon be complete, and those involved in the program will have a complete and easy method set to run a quick-look analysis of the ASARG captive flight test results.

VII. Other Experiences

I would like to express my gratitude to the people of WL/MNGA for many of the other experiences of my summer. I attended many technical briefings, including a seminar on the template generation process for ASARG. I worked with the logistics personnel to set up the calibration array of top hat reflectors on one of Eglin's ranges. I attended a pre-mission planning session, where the flight plan for one of the missions was discussed. Finally, the summer has presented me with an opportunity to conduct independent study and a valuable experience in a mature work environment that will affect future jobs. Although my career interests do not fall within a technical field, I know that this first experience of government work and the exposure to advanced technology has had a tremendous effect on me and my decisions about my future.

Works Cited

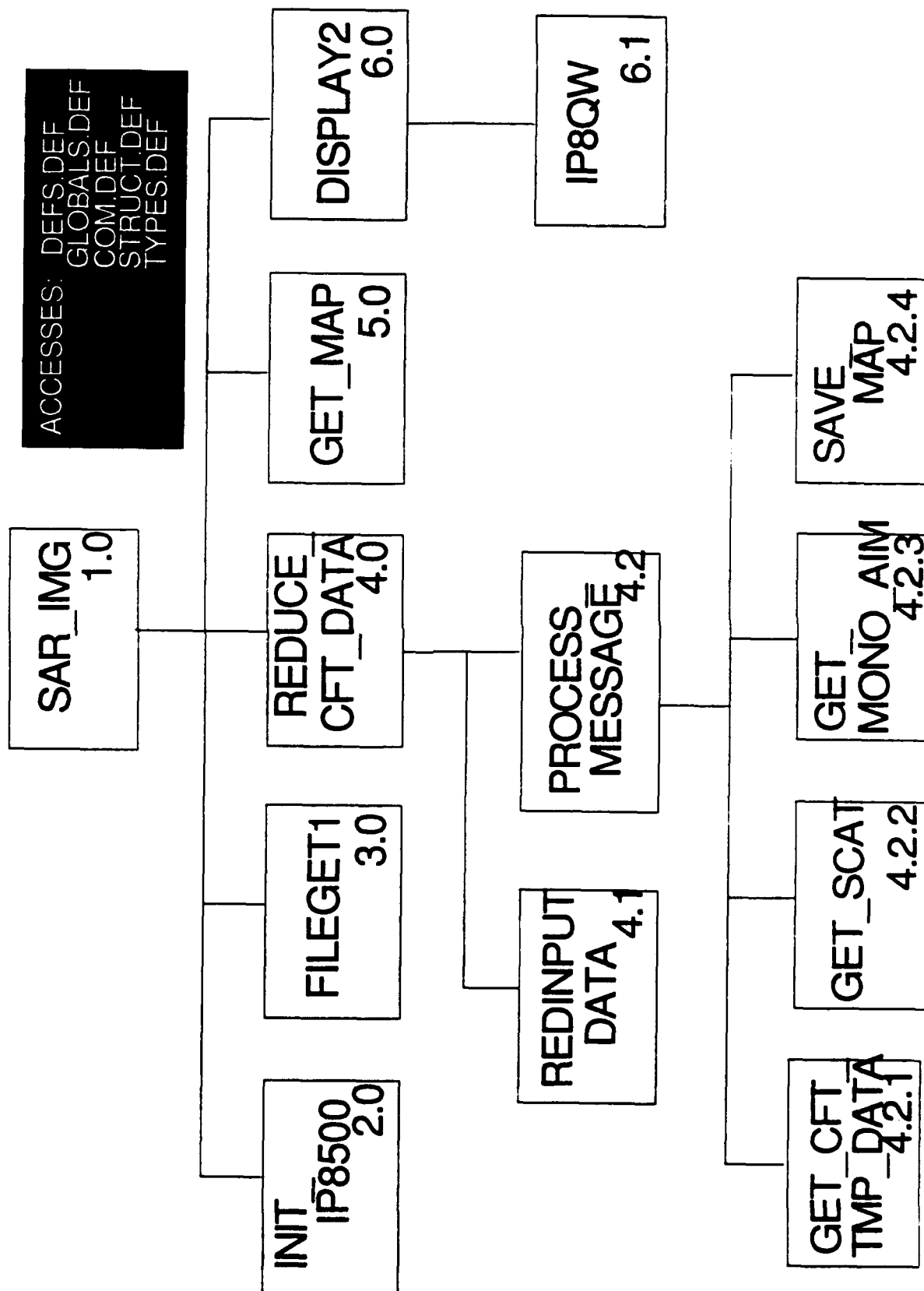
Hovanessian, S.A. Introduction To Synthetic Array and Imaging Radars.

Dedham, MA: Artech House, 1976.

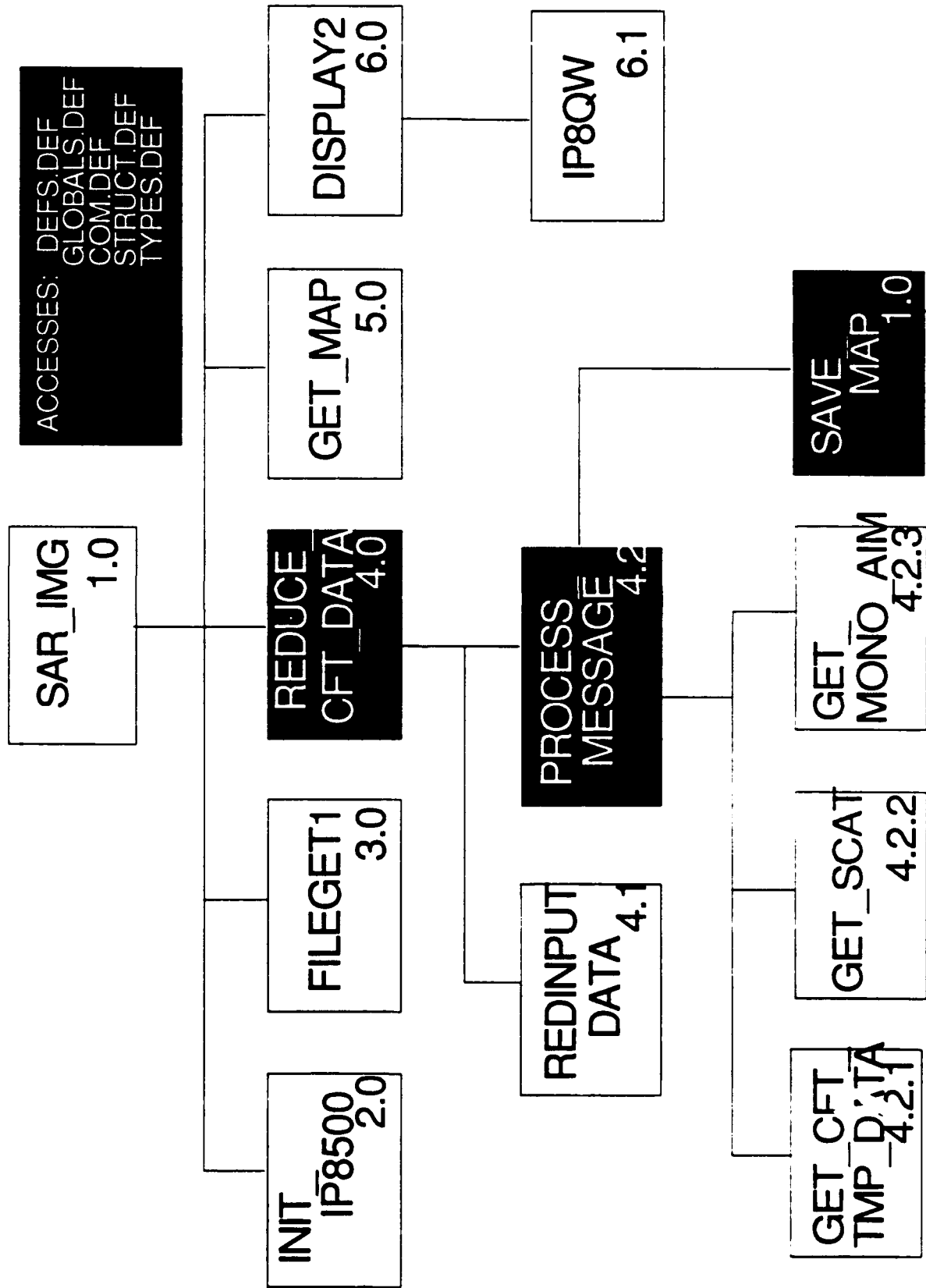
Hovanessian, S.A. Radar Systems Design and Analysis. Dedham, MA:

Artech House, 1976.

APPENDIX 1: SAR_IMG.FOR PROGRAM CHART



APPENDIX 2: SAR_IMG1.FOR PROGRAM CHART



APPENDIX 3: USER INSTRUCTION MANUAL

I OPERATOR INSTRUCTIONS

A VIDEO FRAME GRABBER

1. Turn on power to video equipment. The switch is found on the rack underneath the U-matic Recorder. Make sure that the control box has the following lights on :
IP-8500 Input : VHS, Recorder, Input
IP-8500 Output: Monitor, Comp Gen RgB,
Comp Gen Comp
2. Press the EJECT button and insert tape in slot of Panasonic NV-8500 Video Cassette Recorder.
3. Log-in to IPL on a Digital VT340.
4. Enter DIGIT to run program to grab the video frame by entering:

DIGIT

5. Select DIGITIZER from the menu. Choose the color monitor to be used:
SC01: Tektronix 690SR, left of Video Equipment
SC02: Gould IP-8500, right of Video Equipment
6. Push PLAY on the VCR and the tape will start running on both the TV and the selected monitor.
7. Type 'S' on the keyboard of the VT340 to freeze a frame on the color monitor. Push RETURN to go back to having the tape run on the color monitor. To save the frame on the screen, type another 'S' and push RETURN. Choose EXIT from the menu to leave DIGIT.
8. Press STOP on the VCR to stop playing the tape and then REWIND to return it to the beginning.
9. Load the SAVE85 program by entering:

SET COMMAND SYSS\$SYSTEM:SAVE85

10. Load the LOAD85 program by entering:

SET COMMAND SYSS\$SYSTEM:LOAD85

11. Connect the VT340 with the correct monitor by entering:

ASSIGN EPA0 IP0 (for SC01)
ASSIGN EPA0 IP1
cr
ASSIGN EPA1 IP0 (for SC02)
ASSIGN EPA1 IP1

12. Save the frame captured on the monitor by entering:

SAVE85/COLOR [filename] 0

13. The frame can now be recaptured on the monitors by entering:

LOAD85/COLOR [filename] 0

NOTE: The video frame grabber changes the frequency of the monitor that is not used. In order to use this monitor, you must connect the VT340 to it and enter:

LMR 2 3 1280

B IMAGE HIGHLIGHTING PROGRAM

1. Log-in to the IPL from a VT340.
2. Connect the VT340 to a color monitor by entering:

ASSIGN EPA0 IP0 (for SC01)

ASSIGN EPA0 IP1

or

ASSIGN EPA1 IP0 (for SC02)

ASSIGN EPA1 IP1

3. Initialize the system by entering:

SYSINT

CUROFF

4. Display the image to be highlighted on the screen by entering:

LOAD85/COLOR [filename] 0

5. Run the highlighting program by entering:

R DRAW

6. Make sure the buttons on the cursor control box are in the following positions:

CURSOR #1: ON
CURSOR #2: ON
TRACK : OFF

7. Use joystick on box to place cursor in the desired position and press ENTER to start line.

8. Move cursor to next location and press ENTER. Continue until line is complete.
9. When done with line, press the TRACK switch ON and then OFF again. If you want to start another line, enter a 'Y.' If you want to quit, enter a 'N.'
10. Continue with steps 7-9 until complete.

C SAR IMAGE DISPLAY

1. Log-in to the IPL on the VT340.
2. Connect the VT340 to the correct monitor by entering:

```

        ASSIGN EPA0 IP0 (for SC01)
        ASSIGN EPA0 IP
              or
        ASSIGN EPA1 IP0 (for SC02)
        ASSIGN EPA1 IP1

```

3. Initialize the system by entering:

```

        SYSINT
        CUROFF

```

4. Run the SAR_IMAGE program by entering:

```

        R SAR_IMG

```

5. Type in name of data file when requested (e.g. F24_5). If you just hit RETURN, the program will ask if you want to quit without displaying the image.
6. Choose which track (1-12) when prompted.

D SAR BACKGROUND REMOVAL

1. Display the image on the monitor that is easiest to work with.
2. Run LIPS, an image processing program:

```

        R UTILITY:[GRAPHICS.LIPSEX]LIPS

```

Enter the number for the monitor that the image is displayed on (0 for SC01 and 1 for SC02).

3. Clear the other color channels so that only one is being worked on by entering:

```

        CLR CH 1
        CLR CH 2

```


4. Start the program which will remove the background by entering:

ITDRAW MD 2

5. Use the control box to place the cursor in the top left hand corner if you wish to remove the darker colors or the top right hand corner if you wish to remove the lighter colors. Press 'D' on the keyboard to toggle to draw. Move cursor towards the center until the background is white. Hit RETURN to exit ITDRAW
6. Enter the following commands to copy the color table to channel 0 and then copy channel 0 to channel 1:

COPY LU 0
COPY CH 1 0

7. To clear the ITDRAW box, enter:

CLR CH 3

8. To show only channel 0, enter:

M0

9. Exit LIPS by entering:

EX

10. Save image using EXTRACT by entering:

EXTRACT

11. Initialize the monitor before retrieving the image by entering:

SYSINT
CUROFF

11. Display image in color again by using DISPLAY and running the program COLOR by entering:

DISPLAY
R COLOR

E IMAGE PRINTING

1. Log-in to the IPL from a VT340.
2. Connect the VT340 the Gould IP-8500 (SC02) by entering:

ASSIGN EPA1 IP0
ASSIGN EPA1 IP1

3. Initialize the system by entering:

SYSINT
CUROFF

4. Load the LOAD85 program by entering:

SET COMMAND SYSS\$SYSTEM:LOAD85

5. Display the image to be printed on the screen by entering:

LOAD85/COLOR [filename] 0

6. Press COPY on the TOYO TPG-4300 and the PRINT light will start blinking. The printer will do several overlays of different colors before finishing. When the page is completely ejected, remove the page using the cutter to the right of the carriage return.

APPENDIX 4: EQUIPMENT AND SOFTWARE DESCRIPTION

OVERVIEW

The process outlined in this manual is to be used to examine data from Captive Flight testing during Phase III of the Autonomous Synthetic Aperture Radar Guidance (ASARG) Program. During the testing, the guidance system will be connected to a video camera which will be directed by the aimpoint of the SAR. This process is used to compare images, SAR and video, from the same flight. A frame from the video film is digitized and then printed on hardcopy. The corresponding SAR track is expanded to fill the entire screen. Once the background has been removed by image processing software, the SAR image is transferred to transparency and compared to the video hardcopy. Software provided by TEAS was modified to provide the SAR image while various software and equipment in the Image Processing Lab was used to work with the different images.

EQUIPMENT

Panasonic AT-H190G Color Video Monitor
Panasonic NV-8500 Video Cassette Recorder
Digital VT340
Tektronix 690SR Color Monitor (SC01, EPA0)
Gould IP-8500 Color Monitor (SC02, EPA1)
TOYO TPG-4300 Color Printer

SOFTWARE

A SAR_IMG

This program initializes a Gould IP-8500 (or Tektronix 690SR) Color Monitor to prepare it for the display of a SAR image. After getting the name of the file from the user, it will reduce the data in the file and display the SAR image for the track chosen by the user.

B INIT_IP8500

This subroutine has been designed for use with the Gould IP-8500 Color Monitor but can also be used for the Tektronix 690SR. It initializes the monitor by first clearing the screen and then loading the various tables and defaults needed to display an image.

C FILEGET1

This subroutine examines the file name inputted by the user and adds a ".dat" extension if none is provided. It attempts to retrieve and open this file, giving error messages if unable to find or open it. Once the file is successfully opened, the routine will inform the user.

D REDUCE_CFT_DATA

This subroutine is responsible for reducing the SAR-Seeker System's raw data files for Phase 2 of the Captive flight testing. The format for the Phase 3 data files will be different so this module will either have to be modified or replaced before the program can be used for Phase 3 data. The subroutine initializes and enables the variables and flags used in data reduction. It then calls the necessary subroutines needed to process the file for the desired information.

E GET_MAP

This subroutine uses reduced information from the data file to assign the correct color magnitudes to a 1200 X 1200 array in order to display a SAR image.

F DISPLAY2

This subroutine is used to change the array from GET_MAP into the correct format and then display it on a Gould IP-8500 Color Monitor.

G REDINPUTDATA

This subroutine reads in the information from the data file.

H PROCESS_MESSAGE

This subroutine processes messages from the Binary Input File Data and then calls the correct subroutine needed to find the required information.

I GET_CFT_TMP_DATA

This subroutine obtains CFT template information for each update.

J GET_SCAT

This subroutine finds valid scatter positions for monopulse analysis.

K GET_MONO_AIM

This subroutine does monopulse analysis.

L SAVE_MAP

This subroutine retrieves and saves the information necessary for calculating the color magnitudes in the SAR images.

M IP8QW

This subroutine is an internal function for the Gould IP-8500 which transfers the image data to the monitor and displays the image on the screen.

**Low-Cost Coherent-On Receive
Missile Radar**

by

Dennis W. Scott, Jr.

Francisco Arredondo
August 14, 1991
MNGS, Wright Laboratory
Eglin AFB, Florida

LOW-COST COHERENT-ON-RECEIVE MISSILE RADAR

Dennis W. Scott, Jr.

The project assigned to me this summer dealt with the real time, real world testing, research, and development of low-cost coherent-on-receive missile radar. The greater portion of the project was aimed at the use of a Recirculating Optic Delay Link (RODL) in a radar system to act as frequency reference unit. The RODL is a relatively new and untested concept which has received little attention since its conception and development because of a lack of personnel and priority. This summer it was reinstated for the cause of the High School Apprenticeship Program and the need to develop the project. My job was to advance the project to a stage where it may later be resumed for further development by the faculty of Wright Laboratory. A partner was assigned to work and learn with me in this task so that we would be able to help each other through this relatively difficult project. Ultimately, the project involved devising a simple pulsed generating radar system, implementing the recirculating optic delay link in the system, and evaluating/testing the system. By summer's end the system was progressed from nonexistence to a working setup which could determine targets and their ranges through the implementation of this hardly used concept.

INTRODUCTION

I am Dennis Scott, a senior at Niceville Senior High School in Niceville, Florida. This summer I was allowed the honor of working in the High School Apprenticeship Program at Wright Laboratory, Eglin Air Force Base. My research was conducted under the supervision of my mentor, Francisco Arredondo, in the MNGS Division. This division works with, develops, tests, and researches guidance systems for missiles and other armament with emphasis on the implementation of radar and radar technologies. My work and research during this summer has lasted from June 10 through August 23, 1991.

ACKNOWLEDGMENTS

This summer has been a tremendous learning experience and a great deal of fun. Acknowledgments are given to the people who helped make this summer as productive and fun as it was. I would like to thank my mentor, Frank Arredondo, and Johnny Walker for all of their patience in teaching and helping me this summer. I would also like to my partner, Keith Stevenson, and his mentor Roger Smith, for their help and supervision during this summer's project. Acknowledgments go to Robb Brown, Bob Arnold, and Darryl Huddleston for their knowledge, insight, and suggestions; John P. Walker and Bruce Quarles for their assistance in the Millimeter Wave Lab; and Brian Eplett for his interest and moral support. Finally, but very importantly, I would like to express my gratitude towards Wright Laboratory, the administrators of the High School Apprenticeship Program, and Research & Development Laboratories for allowing me the opportunity to participate in the High School Apprenticeship Program this summer.

BACKGROUND

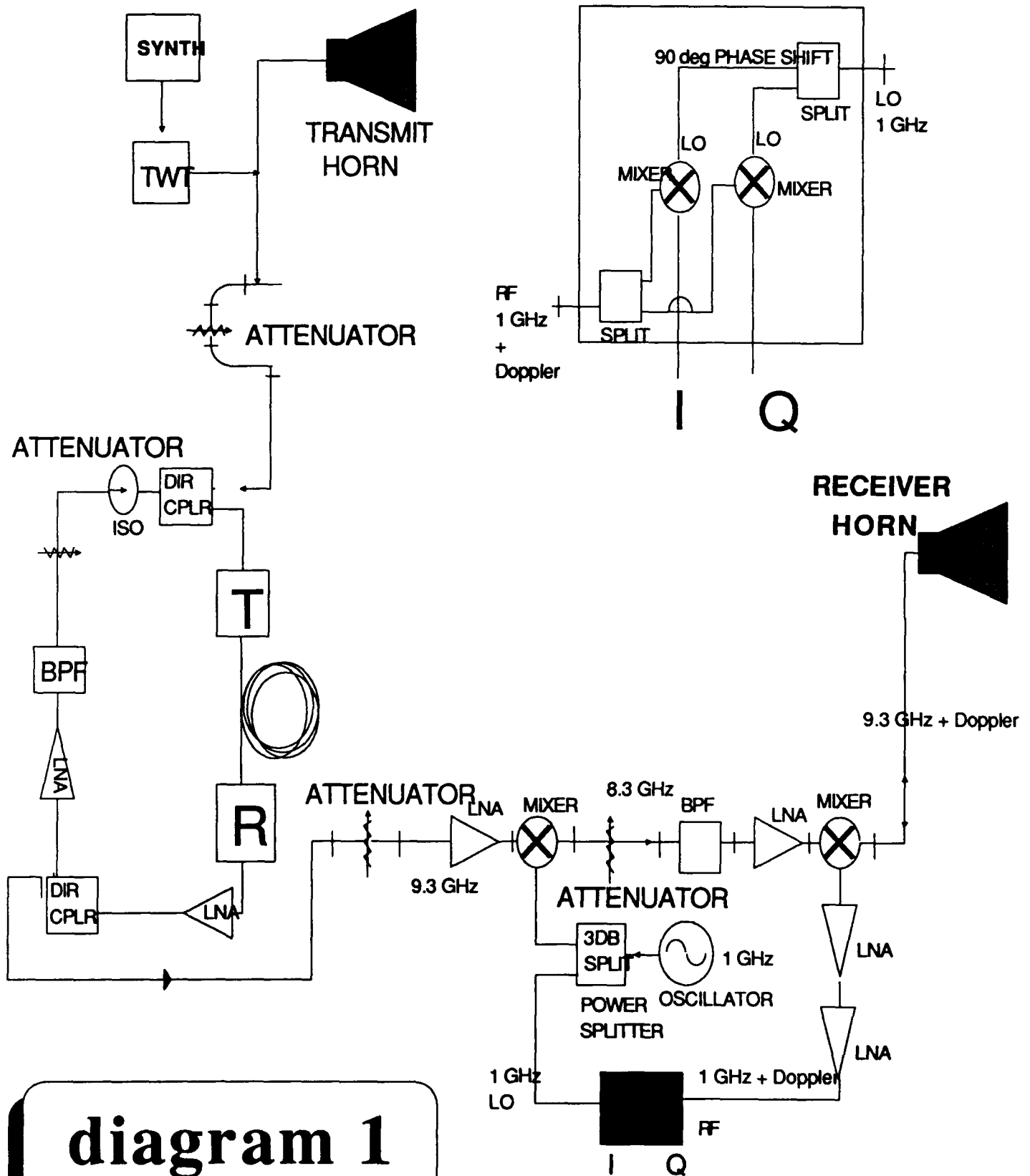
Before reading the explanation of the low-cost coherent-on-receive missile radar project it is somewhat essential to understand some of the basic concepts that deal with the project. The first of these concepts is radar. Radar, or Radio Detection and Ranging, in its very simplest form is basically the transmission of special types of radio waves or pulses and the listening of their echoes. Radar waves or pulses are transmitted through an antenna out into the world--when the signals meet an obstruction, such as a building, an aircraft, or terrain they are reflected--and these reflected signals are then received by an antenna and seen as an image on an oscilloscope, spectrum analyzer, or some other digitizing device. Of course, radar systems are are hardly as simple as this explanation and are in fact much, much more complicated to construct, understand, and in some cases use. Radar is very valuable though in the fact that objects can be not only be detected during day or night in any weather, but information such as range, direction, and velocity can be determined with great accuracy through the use of radar.

It may also be essential to cover some basic terminology that may be used throughout this report. At times there may be references to the term Db. Db is the unit frequently used in the explanation and definition of wave power. It is shown on the spectrum analyzer display as the vertical shifts. The term RODL may also frequently referred to. This stands for recirculating optic delay link, a major component used in my project this summer which will be explained in detail later on in this report.

The research conducted this summer was an offspring of a contracted project with the Hughes Aircraft Company which was initially completed in 1988. The contract called for the Hughes Aircraft Company to develop,

breadboard, demonstrate, and test the new concept of a recirculating fiber-optic delay link which would retain frequency memory in a conventional missile radar system. This would substitute for the conventional frequency reference unit or FRU in a coherent missile radar. Thus, was produced the first Low-Cost Coherent-On-Receive Missile Radar Project. The program was aimed at reducing the the complexity of active RF seekers by simplifying the transmit/receive chain through the use of the RODL. The implementation of such a system would reduce the number of components and subassemblies, and the amount of integration and required testing of the guidance system and, ultimately reducing the cost to produce and maintain the radar.

The concept was assembled by the Hughes Company and tested in a laboratory under totally simulated conditions. Simulators were used in the entirety during the testing stages of the system and concept to create ranges and doppler shifts. Under no circumstances was the system nor the concept tested in a real world situation. This summer Dennis Scott of the High School Apprenticeship Program and Keith Stevenson, an employee in the High School Career Development Program, implemented the fiber optic delay link concept in real time, real world research, testing and development.



RESEARCH

The task of assembling a system to implement the RODL was not a matter of reconstructing what had been sent to Wright Laboratory after the Hughes Company had completed its contract. It became a matter which included constructing a transmitting radar system, integrating the recirculating delay link, debugging and calibrating the system and its subsystems, and testing the implementation of the RODL concept.

The block diagram (diagram 1) displays the radar system designed and built this summer to implement the fiber optic delay link. The diagram displays the signal source, the recirculating optic delay line, the receiver, and a blow up of the I and Q port. The radar signal is produced with the signal generator and TWT (or traveling wave tube). The signal is then split and transmitted through a horn antenna into the world as the second, identical signal is sent to the recirculating optic delay link to be stored. From the RODL the signal then travels to the receiving circuit. The transmitted radar signal is received through a second horn antenna and subjected to a series of frequency mixes to define doppler effects. Doppler shifts may reveal such things as speed and direction of a target through a phase shift created in the I and Q port and shown on an oscilloscope.

The development and construction of this system was achieved through six steps:

- (1) Testing the components,
- (2) Building the recirculating optic delay line,
- (3) Building the receiver,
- (4) Testing transmit power,
- (5) Integrating the transmitter and receiver,
- and (6) Finally evaluating the system.

The tedious work of testing the components involved using a sweep oscillator and spectrum analyzer to determine such things as power and frequency gain or loss, distortions which might occur in specific components, the precision of a certain device, or whether or not a part is working at all. This was done for each of the components you see here on the block diagram and for many other parts which ended up not being used. Building the RODL was quite a task. This time instead of testing individual components we were testing up to a dozen of them at once and trying to make them work together. It took some time and a little amount of headache to get the desired effect out of the system which did not happen to come with instructions. Next came the receiver and the I and Q port. Again, the receiving circuit took quite a bit of work to get the desired effects from the mixes and signal generations. Then the transmit power source was tested and calibrated to get the signal frequency needed with the right amount of Db to power the fiber optic delay line and supporting components, and to create enough of a signal out of the transmit horn to have a strong return to the receive antenna. The transmitting circuit was then integrated with the receiving circuit completing the entire coherent-on-receive radar system. Finally, the entire system was evaluated in order to estimate its potential performance.

To better understand the radar system built this summer and how it functions it is essential to break down the system into its nonintegrated components and define each separately. The explanation will begin at the transmitting end and work toward the reception and generation of the incoming signal. The transmitter consists of a Varian TWT liberated from the Naval Weapons Center at China Lake, California, a Wiltron Swept Frequency Synthesizer made available by the Millimeter Wave Lab of Wright Laboratory, a horn antenna, and an attenuator. A 9.3 GHz pulse with a 60

microsecond PRI, or pulse repetition interval, was first produced in the frequency synthesizer. The signal was sent to the TWT which amplified the signal so that it would be strong enough to travel through the radar setup and also be used as a transmit signal which could be detected. The signal was then split into two identical signals. One was sent through the horn antenna and transmitted into a target area. The second signal was put through an attenuator so that the power level could be further controlled and to prevent a potential burnup of the RODL circuit. The signal was then sent to the the recirculating delay link.

The recirculating delay line consists of a 200 meter fiber optic line, a laser transmitter, a photodiode receiver, directional couplers, an isolator, an attenuator, a 9.3 GHz band pass filter, and amplifiers. The signal from the transmitter enters into the RODL circuit through a directional coupler and then into the laser transmitter. The laser transmitter converts the radar pulse into light and sends it through the fiber optic line. The 200 meter line creates two-thirds of a microsecond delay and is converted back into a radar signal by the photodiode. The signal is then amplified and split in a directional coupler. One of the signals is routed to the receiver circuit as the other is recirculated through the RODL. The recirculated signal is again amplified and sent through a band pass filter which cleans the signal of all noise and clutter gained through circulation and leaving only the 9.3 GHz frequency to continue through the line. From the filter the pulse is received at an attenuator so that control is maintained over the power of the signal. Then, through an isolator the pulse is once again put into a directional coupler to restart its recirculation.

Because of the continuous recirculation of the radar signal through

the RODL, an almost continuous flow of pulses is output to the receiver. The signal, still in its original 9.3 GHz state, is first attenuated to prevent burning up the amplifier. Amplification is necessary to gain enough power in the signal to run the mixer. In the receiving circuit, a Sweep Oscillator is set up to generate a 1 GHz continuous signal. When output from the oscillator the signal is split and sent to the I and Q port and the mixer previously mentioned. In the mixer the 9.3 GHz continuous pulse mixer with the 1 GHz signal created in the oscillator. This mix creates an 8.3 GHz signal which travels to another attenuator and then through an 8.3 GHz band pass filter. The clearly defined 8.3 GHz signal is then sent to another mixer. Meanwhile, an incoming signal produced by a target reflection is received through a second receiving horn antenna. The incoming signal exists as a 9.3 GHz frequency plus the doppler shift produced by moving targets the signal may have encountered. The incoming signal from the horn antenna is mixed in the mixer with the previously stated 8.3 GHz signal. The mixture of the two signals produces a 1 GHz frequency plus any doppler shift present in the signal which had come in from a target area. This 1 GHz plus doppler shift signal enters into the I and Q port along with the 1 GHz signal produced by the sweep oscillator.

The I and Q port consists of two mixers, two splitters, and lines connecting the arrangement. Each of the previously mentioned signals enters into a side of the I and Q port and is split. The 1 GHz signal created in the oscillator is sent through two lines specifically defined to be unequal in lengths so that a 90 degree phase shift is produced between the once identical signals. The split signals from both the oscillator and the receiving antenna are then mixed. The mixing of these two signals cancels out the 1 GHz frequency leaving only the doppler shift which is taken by these two lines to an oscilloscope.

Once the radar system achieved a working status and the debugging was complete it was moved to a penthouse lab atop Wright Laboratory. This enabled clear workspace and open target areas so that the radar system could be tested and calibrated in a real world environment. From the penthouse tests were conducted in the transmitting and receiving needs of the system on a various number of targets.

CONCLUSION

For the endeavors made this summer a working radar system capable of 'seeing' targets and producing their image on a digitizing oscilloscope has been produced. From this image range (or the distance from the transmit/receive point) of a target (moving or stationary) can be determined and a rough estimate of the target's size can be gained. For instance, the radar can tell a building from a stationary truck or a large tree. The radar system is also relatively small. It has easy potential of being reduced in size enough to be installed in a missile. The components used in the setup are also inexpensive enough compared to conventional systems to establish further research and development into this concept.

Further upgrades to the system developed this summer through the research done through the High School Apprenticeship are still needed to make full use and do further tests and development of the coherent-on-receive radar concept. Due to the lack of time during the summer when the apprentices are available project remains incomplete and is passed on to the faculty of Wright Laboratory. Reading doppler shifts off of the setup is still inaccurate and insufficient to the purpose of the system; the timing and digitizing system has yet to be integrated; the setup must be breadboarded to reduce insufficiencies in the system; and more research and development must be done to improve on the unknown or untested factors which still exist.

A premature conclusion to the research done this summer would have to include the validation of the recirculating optic delay link concept in a real world, real time environment. It is also suggested by the findings made that further research on this concept would be well worth while in ultimately reducing the cost to produce and maintain coherent missile radar

systems. Furthermore, it is essential that the project be expanded and carried further to test doppler sensing capabilities as well as any other pertinent radar needs.

LESSONS LEARNED THIS SUMMER

The summer has been quite a learning experience. The amount of applied concepts and research techniques I have been taught surpasses any experience I have ever encountered in school. I was taught and trusted by my mentor and lab personnel to use hardware systems this summer which are not readily available at many major engineering universities. I've gained experience in the use of a Varian TWT, a Wiltron Swept Frequency Synthesizer, Hewlett Packard Sweep Oscillators, Hewlett Packard Digitizing Oscilloscopes, a Hewlett Packard Spectrum Analyzer, an HP System Graphics Display, a LeCroy Digitizing System, and many other pieces of equipment which became tools used on a daily basis. I was also taught the formal concepts of radar, the structure and functions of a radar system and the individual parts which make up that system. I can now identify a coupler from other components such as mixers, or amplifiers and a little more importantly--I know the function of each. And, of course, I now know how to construct a radar system and identify targets when I see them.

I was also taught the concepts of C programming and have been instructed on the use of computer systems and software. I have also been taught new research methods and documentation disciplines which will last throughout my senior year and college career. Most importantly, the responsibility of having a job and fulfilling the responsibilities of doing well in it have been a major education for me.

BIBLIOGRAPHY

Hughes Aircraft Company, Missile Systems Group, Low-Cost Coherent-On-Receive Missile Radar; Technical Reports, final report, report no. COR-8-001, Hughes Reference No. G-1956, Canoga Park, CA, December, 1988.

Digital Signal Processing

Troy Urquhart, WL/MNSI

Final Report: Summer 1991

DIGITAL SIGNAL PROCESSING

TROY URQUHART, WL/MNSI

During the summer of 1991, I worked in the signal processing section of the Guided Interceptor Technology Branch in the Wright Laboratory Armament Directorate on Eglin Air Force Base (WL/MNSI) under contract of Research and Development Laboratories, Inc. (RDL). Working with mentor Captain Allen Andrews and colleague Derek Holland, I was involved in a variety of projects, a summary of which follows.

DIGITAL IMAGE COMPRESSION WITH THE DISCRETE COSINE TRANSFORM

One of our more difficult tasks this summer was that of image compression. There were two fundamental approaches to the development of image compression algorithms: compression using the Discrete Cosine Transform, described in this section, and compression using fractal geometry, described later in this report. Derek and I worked under Rick Wehling on the DCT approach and under Lee Murrer on the 'Barnsley board' fractal approach.

The Discrete Cosine Transform (DCT) can be derived directly from the real portion of the Fourier transform and, like the Fourier, remaps the image data from a domain of intensity to that of frequency. The basis function for the one-dimensional DCT over an array comprised of N elements can be defined as

$$C(0) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f(x)$$

(Eq 1.1a)

$$C(u) = \sqrt{\frac{2}{N}} \sum_{x=0}^{N-1} f(x) \cos \frac{(2x+1)u\pi}{2N}$$

(Eq 1.1b)

where $u=0, 1, 2, \dots, N-1$, and where the inverse function is defined as

$$f(x) = \frac{1}{\sqrt{N}} C(0) + \sqrt{\frac{2}{N}} \sum_{u=1}^{N-1} C(u) \cos \quad (\text{Eq 1.2})$$

for $x=0, 1, 2, \dots, N-1$. However, the DCT's derivation from the Fourier transform is more easily seen by defining the DCT as

$$C(0) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f(x) \quad (\text{Eq 1.3a})$$

$$C(u) = \sqrt{\frac{2}{N}} \operatorname{Re} \left\{ \left[\exp \left(\frac{-j\pi u}{2N} \right) \right] \sum_{x=0}^{2N-1} f(x) \exp \left(\frac{-j\pi ux}{N} \right) \right\} \quad (\text{Eq 1.3b})$$

where $u=1, 2, \dots, N-1$; $f(x)=0$ for $x=N, N+1, \dots, 2N-1$; and $\operatorname{Re}(\)$ denotes the real part of the term enclosed. However, I must admit that definition of the DCT by (Eq 1.3a-b) is extremely unwieldy when programming.

Due to its relatively uniform output, even of very dissimilar images (Fig 1.1a-b), the DCT has served as the basis function for all compression techniques developed in-house to this point.

The first DCT compression algorithm used incorporated a two-dimensional DCT, followed by a primitive block quantization algorithm. The two-dimensional DCT for an array of $N \times N$ elements is derived by performing successive applications of a one-dimensional DCT:

$$C(0,0) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \quad (\text{Eq 1.4a})$$

$$C(u,v) = \frac{1}{2N^3} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \left[\cos(2x+1)u\pi \right] \left[\cos(2y+1)v\pi \right] \quad (\text{Eq 1.4b})$$

for $u, v=1, 2, \dots, N-1$ and

(Eq 1.5)

for $x, y=0, 1, \dots, N-1$. The main drawback of this technique is the loss of speed due to the large summation required for each pixel, a 128×128 image often taking upwards of 20 minutes of transform time on a SUN 4.

Our next attempt at compression using the DCT was derived from the Joint Photographic Experts Group (JPEG) standard for image compression. This technique separates the 128×128 image into 256 separate 8×8 arrays, defining the transformed image and its inverse according to (Eq 1.4-5). Not only does this algorithm execute significantly faster (due to the much smaller summation required--64 points instead of 16384!), but it also produces seemingly ideal input for block quantization (Fig 1.2). However, we found that this method was unable to provide the high 10:1 compression ratios required without significant loss of image data.

At this point, my mentor became actively involved in 'the battle for image compression,' and we developed yet another algorithm for compression. This most recent (and most successful, I might add) compression technique is similar to the JPEG technique in that it uses successive applications of the DCT to sections of the image. However, this algorithm uses successive one-dimensional transformations (Eq 1.1) of 32 pixel segments of the array. (Fig 1.3) The output of this transform is then compressed using block quantization techniques. (For a complete description of the block quantization algorithm used, please see Derek Holland's paper entitled 'Transforms, Image Compression.') At present, this method provides ratios of between 6 and 8:1 without excessive data loss (Fig 1.4a-b).

EVALUATION OF FRACTAL IMAGE COMPRESSION

The realm of fractals comprises a geometry which 'describes many of the

irregular and fragmented patterns' found in nature. As Euclidian geometry is the study of 'perfect' shapes such as circles, lines, and planes, fractal geometry is the study of natural patterns such as coastlines, cloud formations and lightning.

It has been shown that fractals can be useful in the compression of images. The commercial pioneer in this field is Iterated Systems, Inc., a company headed by Michael Barnsley and Alan Sloan. Barnsley and Sloan use a hardware system to find fractal equations describing an image, then store these equations as the compressed image. (I must note that this description of their method is pure hypothesis on my part; Iterated Systems, Inc. has declined to release any of their algorithms to this point.) This technique shows great promise in the world of natural images (compression ratios of well over 10:1 with little or no visible loss--Barnsley even claims ratios of up to 100:1 for some images), but it has yet to be determined if these techniques will be useful in the storage and/or transmission of plume data. At present, I am working with Derek Holland, Captain Allen Andrews, and Lee Murrer in evaluating the ISI Fractal Image Compressor, something which I hope to complete by the end of this week (16 Aug 91). At present, high compression ratios with moderate data loss have been attained (Fig 1.5), but further development and experimentation with image padding still needs to be done before an accurate conclusion concerning data loss can be reached.

I am currently developing a program to be used in the evaluation of the 'Barnsley Board.' This program is designed to place an image in the proper Targa format (an industry standard developed by Truevision, Inc.), perform a compression/uncompression cycle, and store the resulting image. Then, both the original image and the result will be sent across the ethernet to a SUN 4

for analysis.

SIGNAL PROCESSING IN THE SYMBOLIC ALGORITHM DEVELOPMENT ENVIRONMENT (SADE)

The Symbolic Algorithm Development Environment (SADE, pronounced 'Say-Dee') is an environment developed in-house by Captain Allen Andrews specifically for the development of signal processing algorithms. This is done by combining items into processes, and these processes into tasks. First, a few simple definitions. An *item* is a memory storage area, usually referred to as a variable in computer programming; it is defined by using the SADE Item Editor, which determines what type it is (floating point, integer, etc.) and its dimensions (1 value, a 128x128 array, etc.). A *process* typically has an input of several items, performs some calculation using those items, and outputs one or more items; it is created by linking the executable (compiled C or FORTRAN) code to SADE through the Process Editor. A *task* is a combination of these processes (or perhaps other tasks) to form a complete algorithm.

One of my projects for this summer was to develop several signal processing operators for use in testing and demonstrating SADE. A simple description of the edge operators: All of the nonlinear edge detectors I coded use either a 2x2 or 3x3 window. This window is scanned across each pixel in the image, a calculation is made, and the result is placed in a result array. Thus, the new pixel value is a value based largely on relationship to its neighboring pixels (Fig 1.6).

The incorporation of these into SADE is a fairly simple matter, due largely to the incredible flexibility of the environment. Once processes have been defined in SADE, their icons are dragged into the current task with the

mouse and connected. Thus, a complete algorithm can be created quickly and is easily readable, much like a flowchart.

THANKS, ETC.

Just a few people I'd like to thank: Captain Allen Andrews, for being a great mentor and for all your C advice; Derek Holland, best of luck at Rice--it's been a great two summers!; Bob LeBeau, for whatever--I just thought I ought to include your name; Paul McCarley, for your interest and your help (when you weren't TDY, that is!); Rick Wehling, for taking a couple of lowly HSAPs seriously; Lee Murrer, for putting me to work on the Barnsley board, and for some really great tech support every time I crashed the 386 so badly I couldn't fix it myself; Greg VanWiggeren, for being such a great friend; Bobby Stockbridge, for all the help with the hardware; Dorothy Williams, for not making me 'do the Hough' transform, and for your admiration of my computer crashes; Mickie Phipps, for giving me the wonderful opportunity of attending the RI SPPD class; Victoria Franques, for blaming Derek and I for killing Pluto (You know, now you can't call me a 'high school kid' anymore); Dennis Garbo, for the SUN help; Mike Couvillion, the computer guru of MNSI, for all his help--'A Star is Born' (ugh!); John Provine, the 'big boss'; Mike from KHILS, for the help with the new drive and the ethernet setup; Don Harrison, for heading this program and making it what it is today; Research and Development, Inc., for that paycheck every couple of weeks, and for making this program possible; Ron Rapp, I won't forget to talk to Don Prathouse and Azahm -- 'coulomb cap,' right?; Starla Christakos, for taking a year off--no, seriously, for getting me off to a great start last summer; Cliff Alexander, IS Fizzix Fun?; everybody else at MNSI, for always being willing to take time out of your day to share your knowledge and technical expertise; Tom Clancy, for finally publishing the new book, The Sum of All Fears; Mario Puzo, author of The Godfather and Fools Die; Joseph Heller, for writing Something Happened;

Andrew Lloyd Webber ('The Phantom of the Opera'), Morgan Creek (the 'Robin Hood' soundtrack), Journey, Sting, The Eagles, Bob Seger (& the Silver Bullet Band), The Police, RUSH ('Time Stand Still'), Harry Connick, Jr. (incredible concert in N'Orleans), Frank Sinatra, Chicago, The Cure, Mozart, New Order, U2 ('Am I buggin' ya?'), Garth Brooks, Jimmy Buffet ('Why don't we...?'), Led Zeppelin (Thanks be to Paul), Hank Williams, Jr. (INTERESTING concert in P'Cola), Simple Minds, UB40, Beethoven, Bryan Adams ('Everything I Do'), Oscar Peterson (with Ray Brown & Herb Ellis), and R.E.M. (yes, it does stand for Rapid Eye Movement), for the musical entertainment that helped get me through the summer.

REFERENCES

If I were to include every book and paper from which I learned, confirmed, or refreshed my memory of information this summer, this section of my paper would run *ad infinitum*. Therefore, I have included only those works upon which a significant portion of my projects were based on. I would also like to note that while I gained a large amount of knowledge from books, it will never compare to that which I gained by talking with people; thus, I must consider my primary reference the people I worked with before I mention any publication.

Barnsley, Michael F. et al. 'A Better Way to Compress Images.' 1988.

Devaney, Robert L. et al. Chaos and Fractals. American Mathematical Society. Providence: 1989.

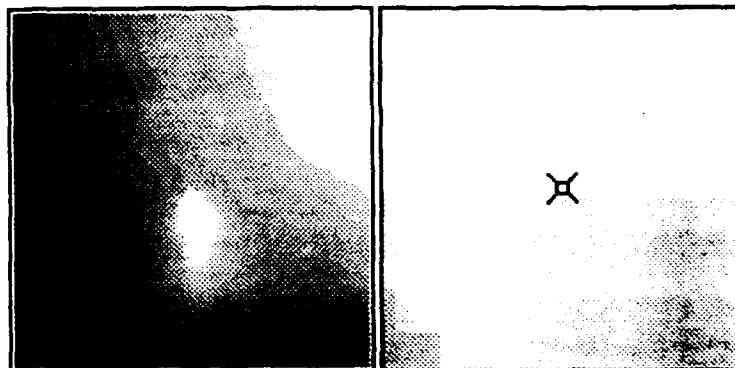
Feder, Jens. Fractals. Plenum. New York: 1988.

Gonzalez, Rafael C. et al. Digital Image Processing (1 and 2 ed). Addison-Wesley. Reading, MA: 1977, 1987.

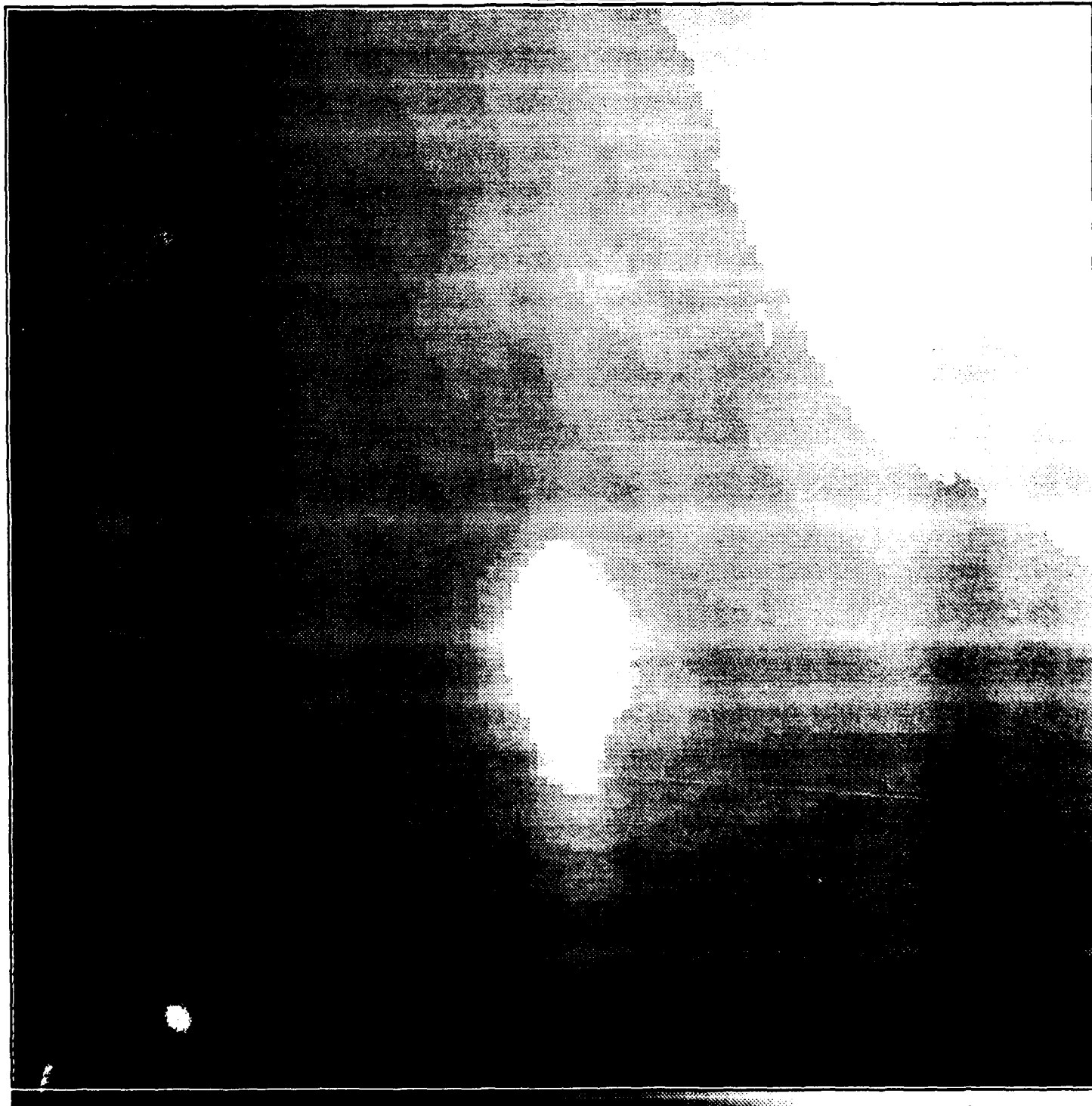
Mandelbrot, Benoit B. The Fractal Geometry of Nature. Freeman. New York: 1977.

Pratt, William K. Digital Image Processing. Wiley-Interscience. New York: 1978.

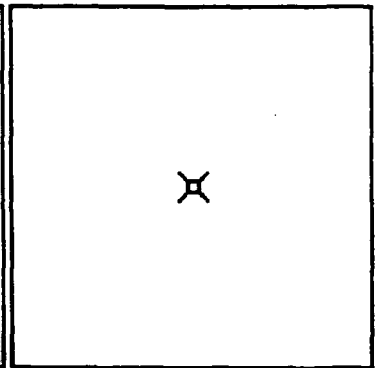
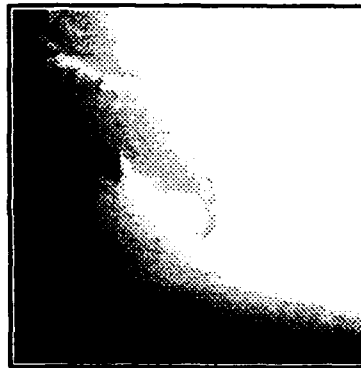
file: picture.4
dir: .



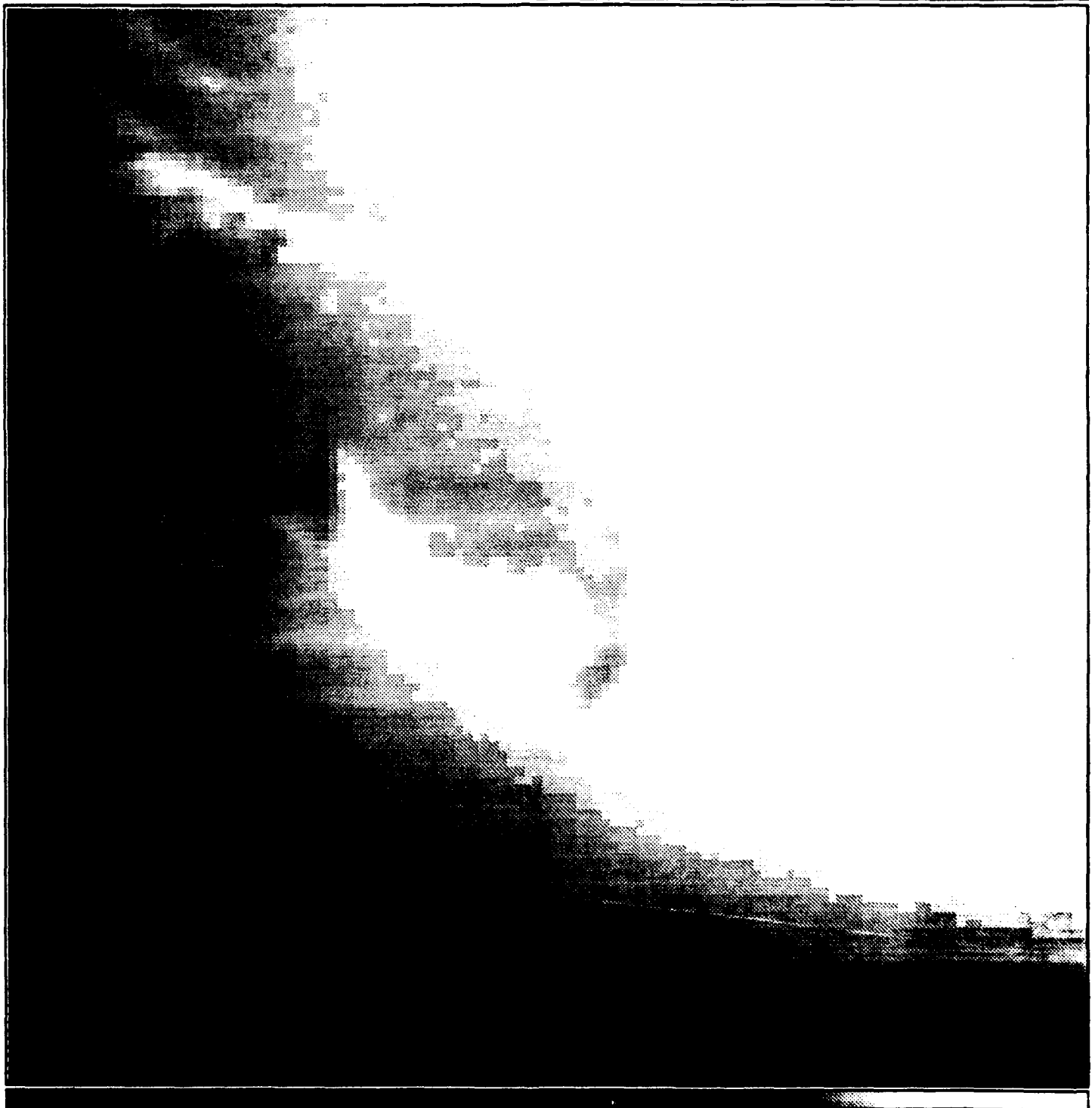
82.2 128.4



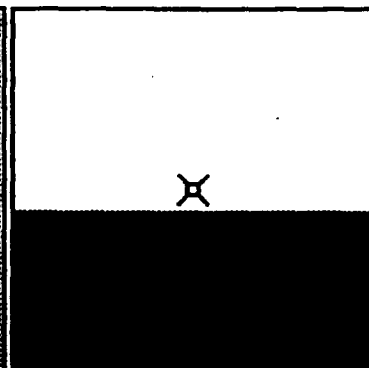
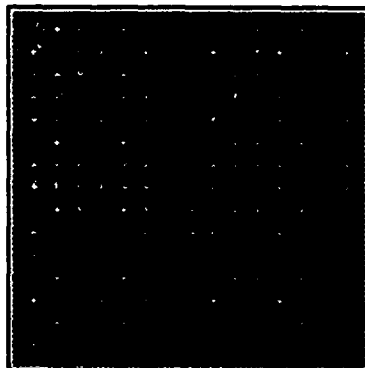
file: liberty128.dct
dir: .



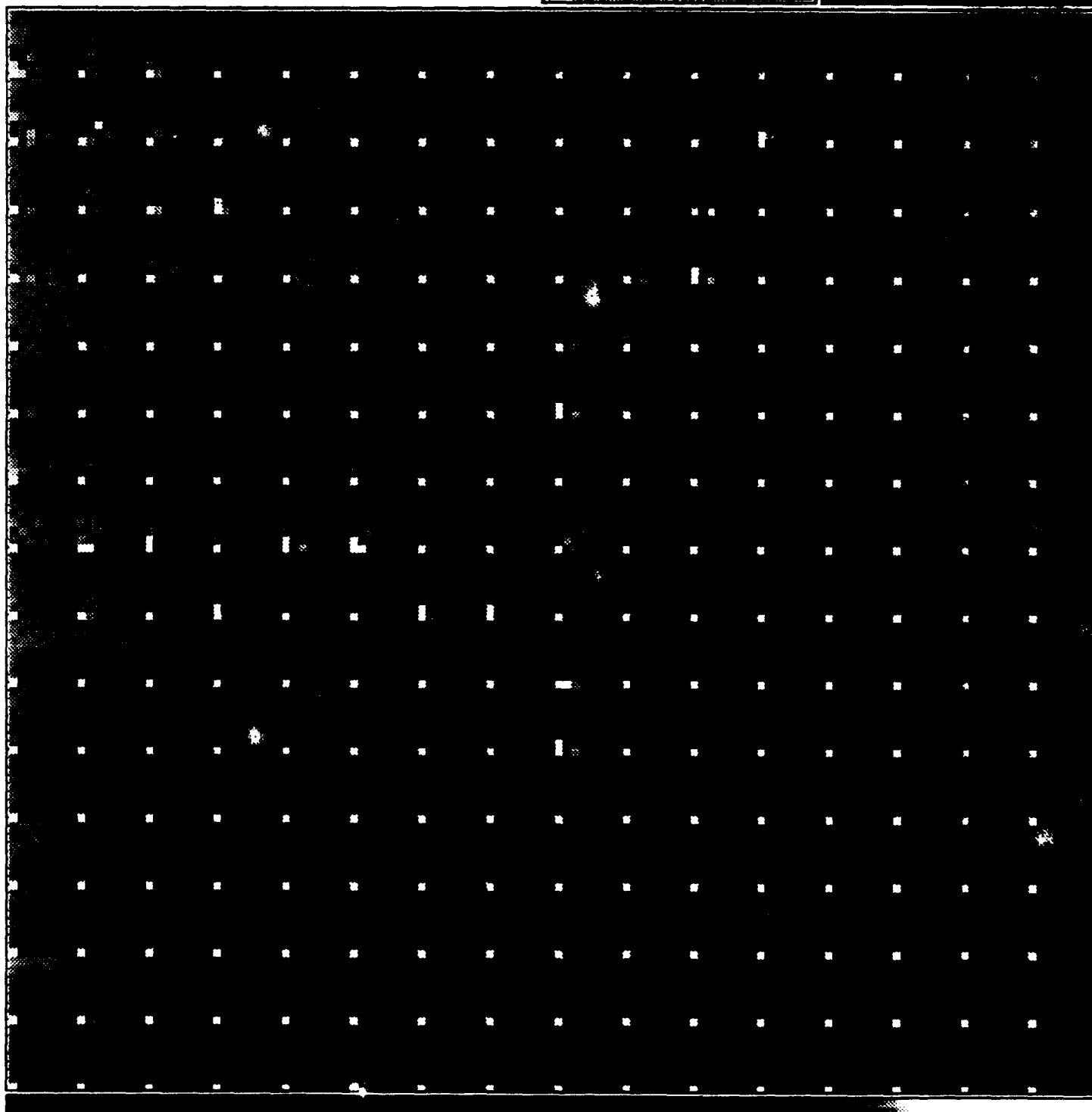
79.8 128.5



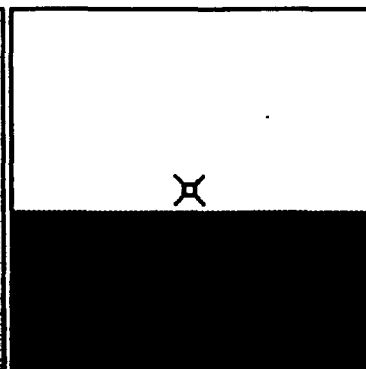
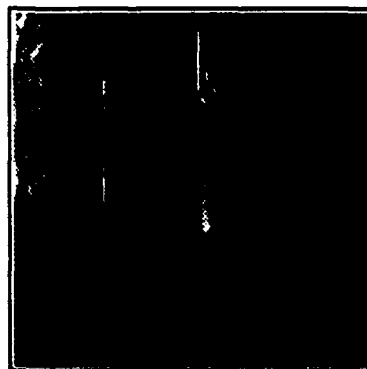
file: liberty128.out
dir: .



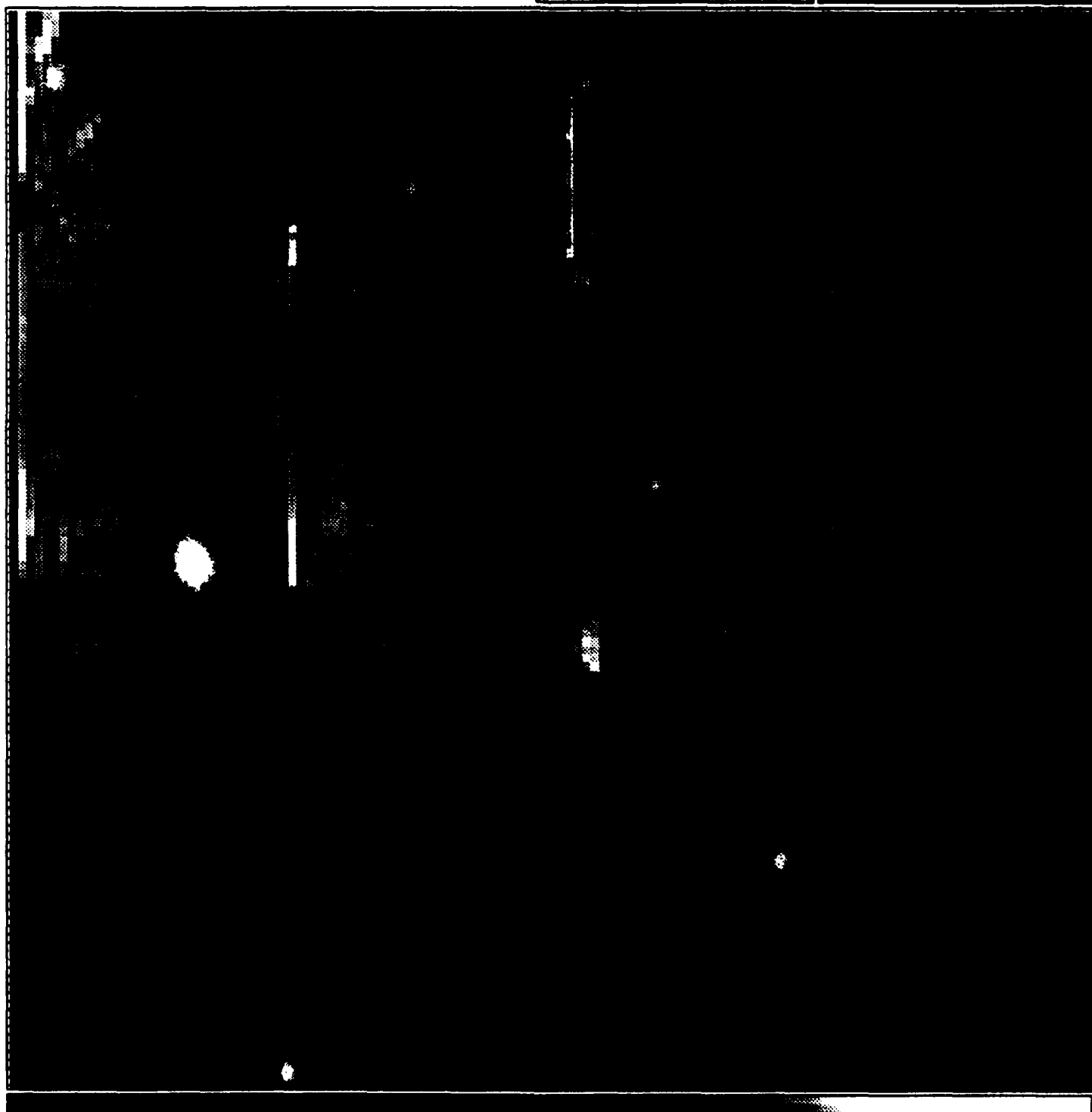
77.0 109.0 3



file: i2.dct
dir: .

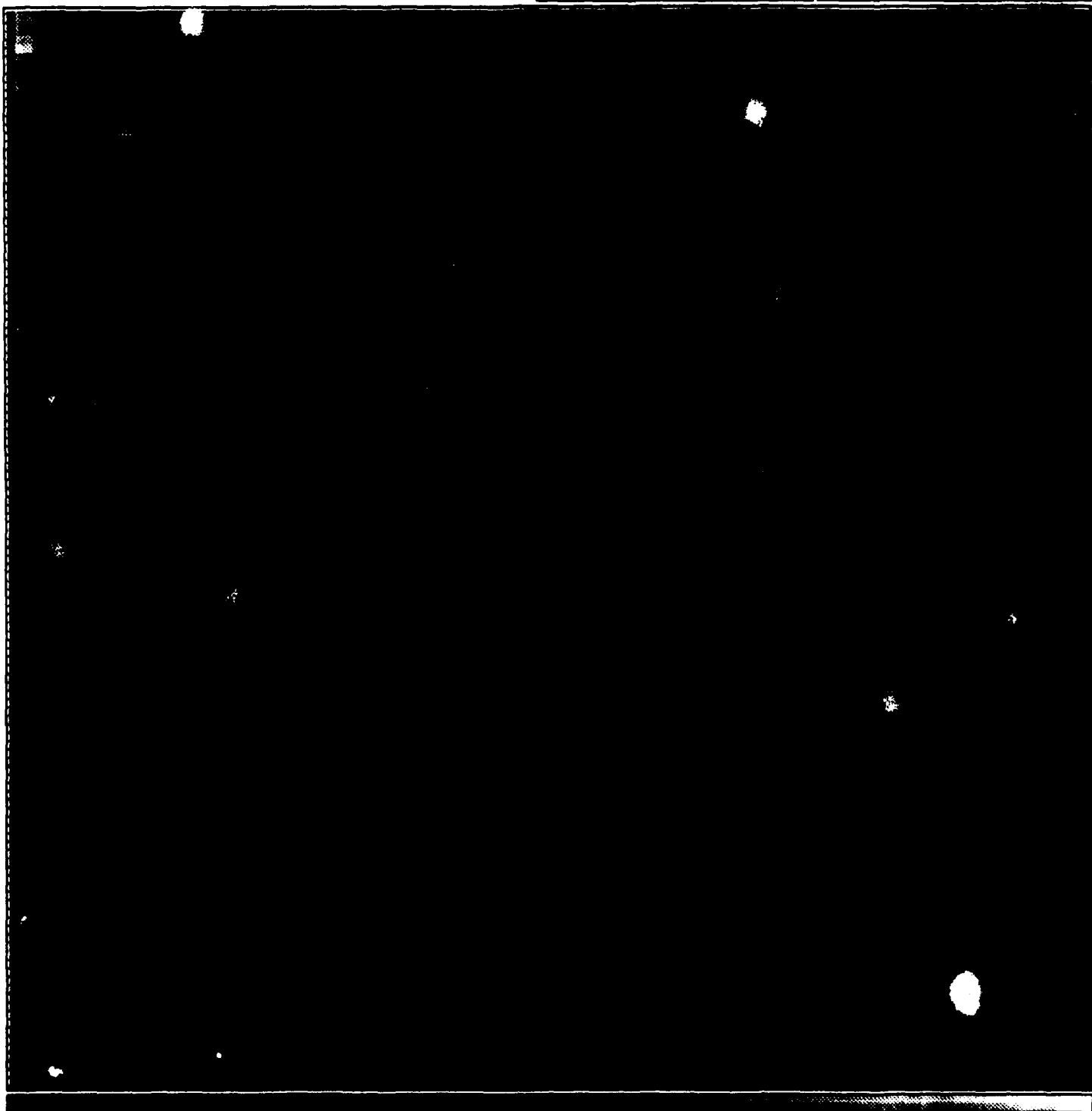


119.2 129.1

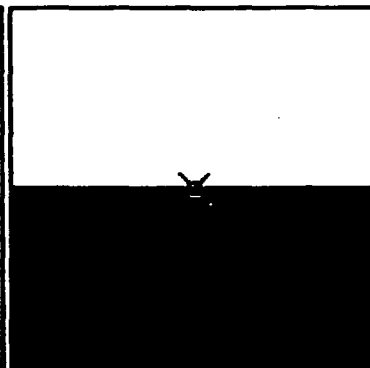
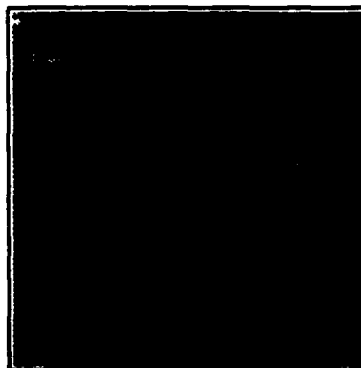


file: liberty128.img
dir: .

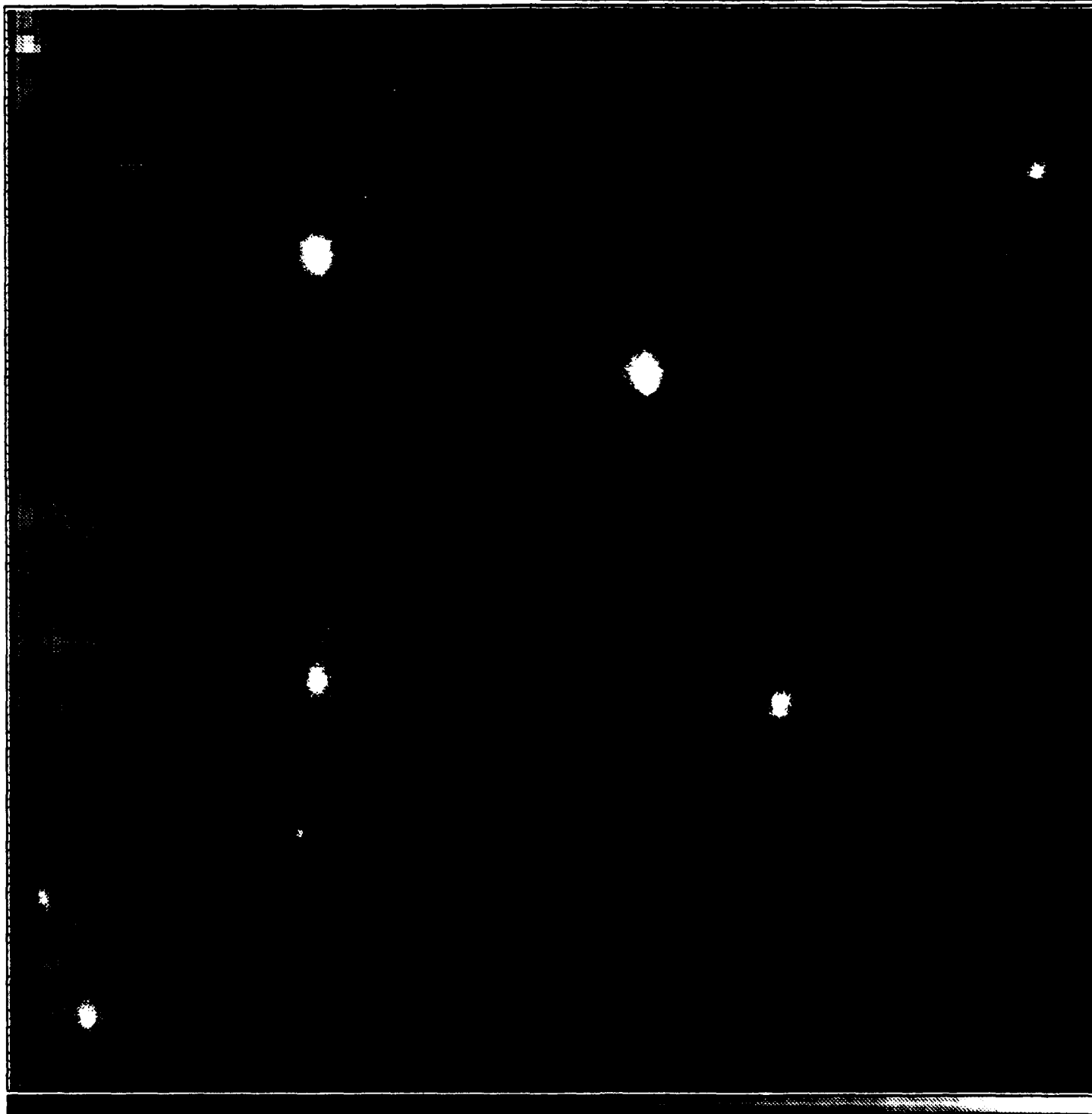
File: 128.0



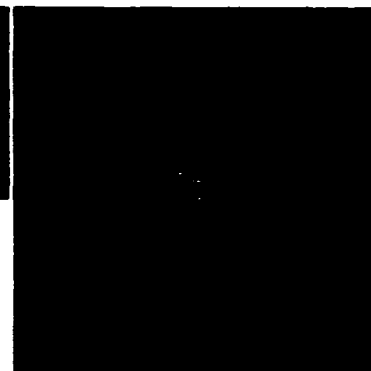
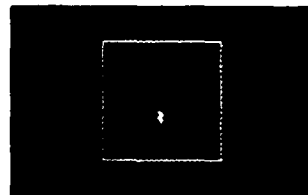
file: i2.img
dir: .



81.0 128.5



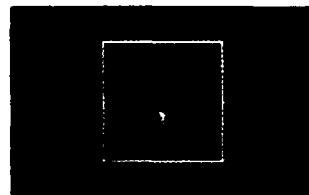
file: f00001.tga
dir: .



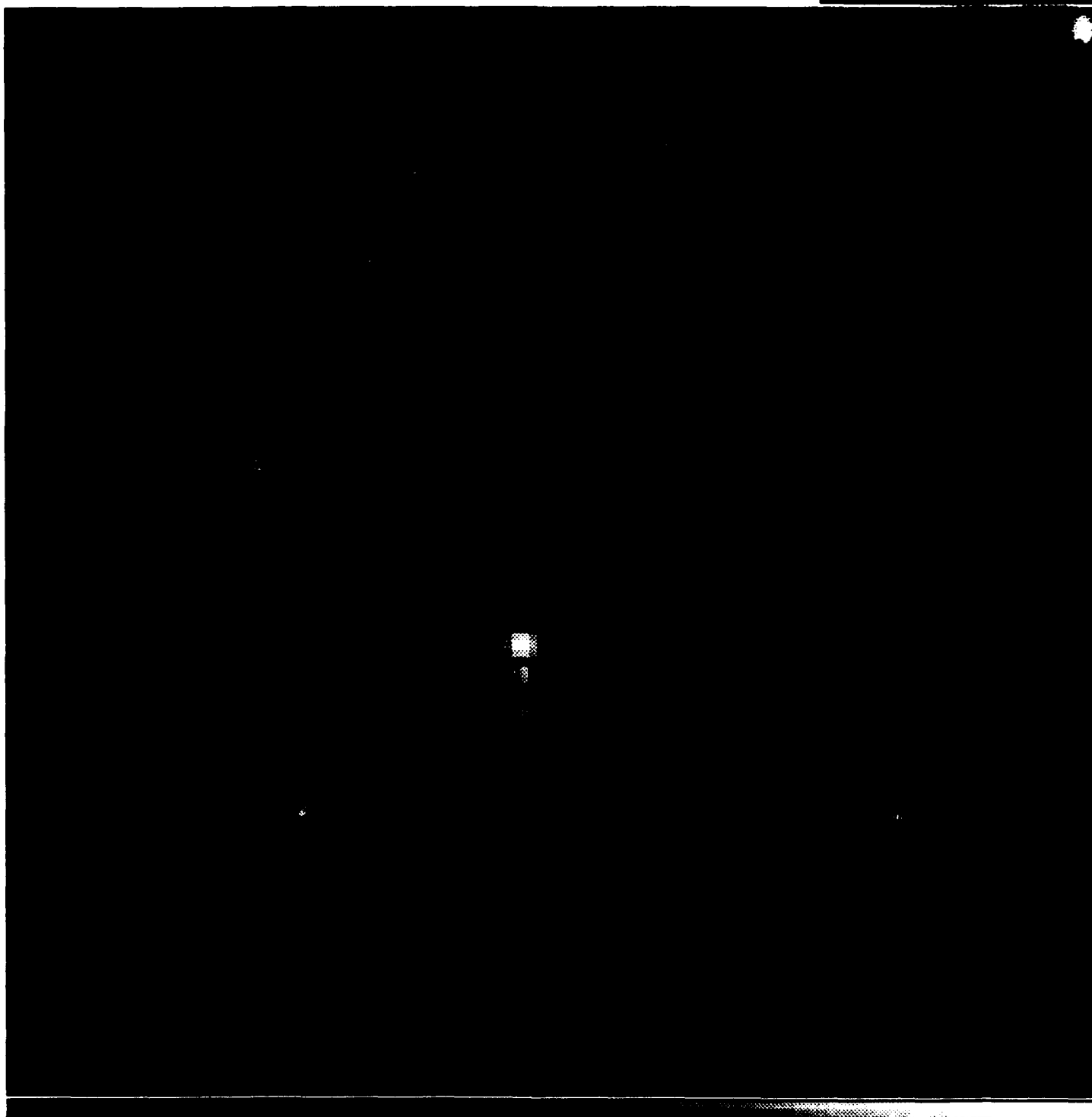
180.2 155.0 13



file: f00001.out
dir: .

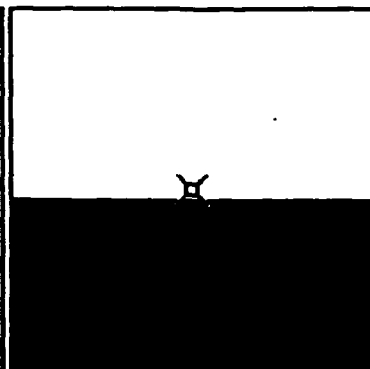


184.2 185.0 186.0



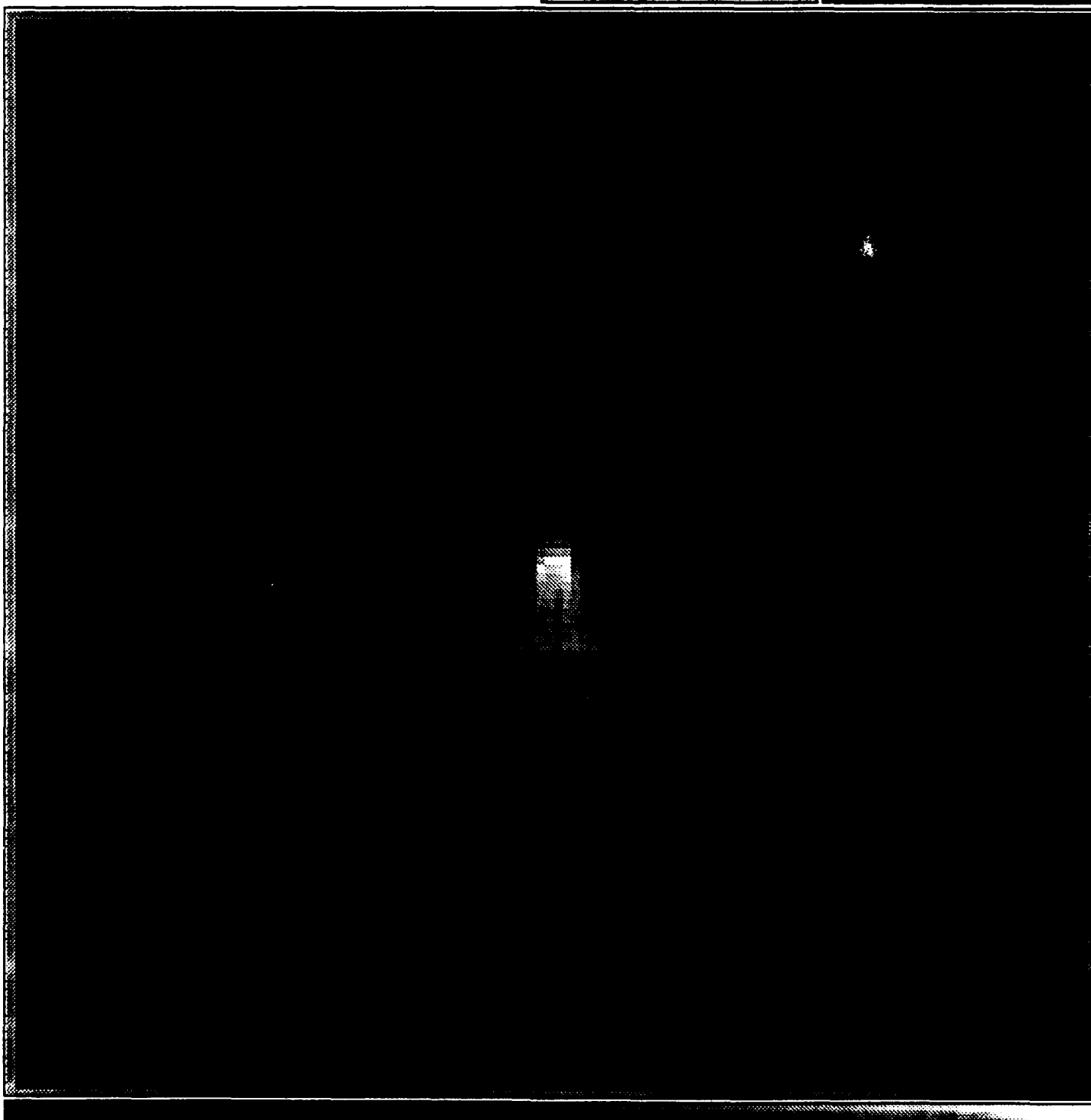
001.IMG

72.2 128.6 1



file: Fsob001.img
dir: .

61.5 128.8



Interior Ballistics Applications in Aerospace Research

WL/MNAA

Gregory D. VanWiggeren

Mentor: Capt. Mike Valentino

July 19, 1991

I. Introduction

This summer is my second in the Armament Directorate Apprenticeship Program. This year, like the previous year, I learned a great deal. I gained many valuable experiences, made new friends, had fun, and hopefully, contributed to the laboratory. Now, as my summer employment is ending, I must launch my college career, and I know the experiences gained these last two summers will benefit my college career and ultimately have a positive affect on my choice of occupation.

My summer project this year involved modifying a light gas gun simulation program. Before beginning work on my project, I had not used BASIC, the program's language, since seventh grade. The complexity of the program made it difficult to find and correct logic errors, and in the end, the original source code was left almost unaltered in order to preserve the same flow of operations. I hesitated to create and use new sequential input and output files and other programming techniques because I had no experience with them, but the program modification forced me to learn new aspects of programming, and I feel it was a good experience for me.

II. Acknowledgments

I first want to thank my mentor, Captain Mike Valentino, for giving me the opportunity to work in the Aeroballistics Section. Without his expertise, his guidance, and especially his patience, these last two summers and everything I learned here would not have been possible. He answered all my questions and often took time out of an already busy schedule to help me with general problems, or explain something about his own college experiences.

The other engineers also helped me a great deal. Dr. Bob Courter answered my probing questions about the light gas gun and helped with many other things. Lt. Rico Vitale and Russ Adelgren shared their expertise -also. If I had a question, they would always try to help; Rico even took me to play racquetball. Lt. Mike Stephens gave me some pointers on my briefing, and Heinz Macker, an exchange engineer taught me about Germany. Kirk Vanden, over in CFD, bent over backwards to help me get viewgraphs for my briefing and the pictures you see in attachment one.

Lastly, I must thank Don Harrison for taking care of all the apprentices, and Dr. Norm Klausutis for his support of the program. This program really is a great opportunity for everyone involved.

III. Background

Before attempting to explain my summer research project, some background should be provided about the Aerodynamics Branch and, in particular, the Aeroballistics Section. The Aerodynamics Branch of the Armament Directorate consists of two sections, the Computational Fluid Dynamics Section, and the Aeroballistics Section. The CFD branch calculates fluid flow predictions for various missile shapes and configurations using a CRAY Y-MP super computer and various other high-power scientific computers. The CFD branch offered me a chance to model a test case using simple prediction techniques, but unfortunately, my early departure date does not leave me enough time to undertake the project. The pictures in attachment one provide examples of the type of work the CFD section does. The computer-simulated paths of trillions of different particles are traced over the missile's surface according to equations for fluid flow. The particles are called floating points and the calculated paths are called floating point operations, abbreviated as "flops". The images shown in attachment one required 5-15 trillion flops. The results of these flops are compiled and graphically represented in images like these. The ribbon-like structures represent vortices. The projectile shown has a simulated pitch of thirty-eight degrees.

The Aeroballistics Section is comprised of the support staff of two test facilities and a group of engineers. Aeroballistics is a field concerned with bullets, rockets, missiles, and other projectiles. The section's mission is to gather data about the aerodynamic properties of projectiles for all branches of the military, contractors, and some in-house work.

The Ballistic Experimentation Facility (BEF) uses guns to shoot projectiles into a sandtrap. The projectile models are usually shot from either a twenty, thirty, or forty millimeter powder gun or light gas gun. X-ray cameras, and high speed video equipment are used to determine the model's structural integrity. Before the shots, paper yaw cards, which help determine the orientation of the projectile, are placed along the predicted trajectory of the model. The angle of attack and other data can be obtained from the size and shape of holes torn in the paper. If the model flies well consistently it may be brought to the Aeroballistic Research Facility (ARF) for more detailed testing.

The ARF provides the engineers with much more complete and precise information about the projectile's flight. The ARF, unlike the BEF, is enclosed and well instrumented. Figure 1 shows the facility layout. One of various guns, ranging up to 76mm, shoots the model down the 207 meter range. Cameras placed along the range, as shown in figure 2, record the projectile's position, orientation, and speed. The projectile continues downrange until it reaches a "bullet catcher" and is demolished on impact. A permanent bullet catcher is placed at the end of the range, but a mobile catcher can be moved closer to ensure that less stable missiles are caught before they damage the facility. The pictures obtained, seen in figure 3, are digitized and subsequently analyzed by the engineers to determine the aerodynamic properties of the model. These aerodynamic properties are obtained through regressive techniques by comparing the experimental data with a simulated trajectory, as shown in figure 4. My summer project last year involved this type of data reduction and analysis.

Figure 1

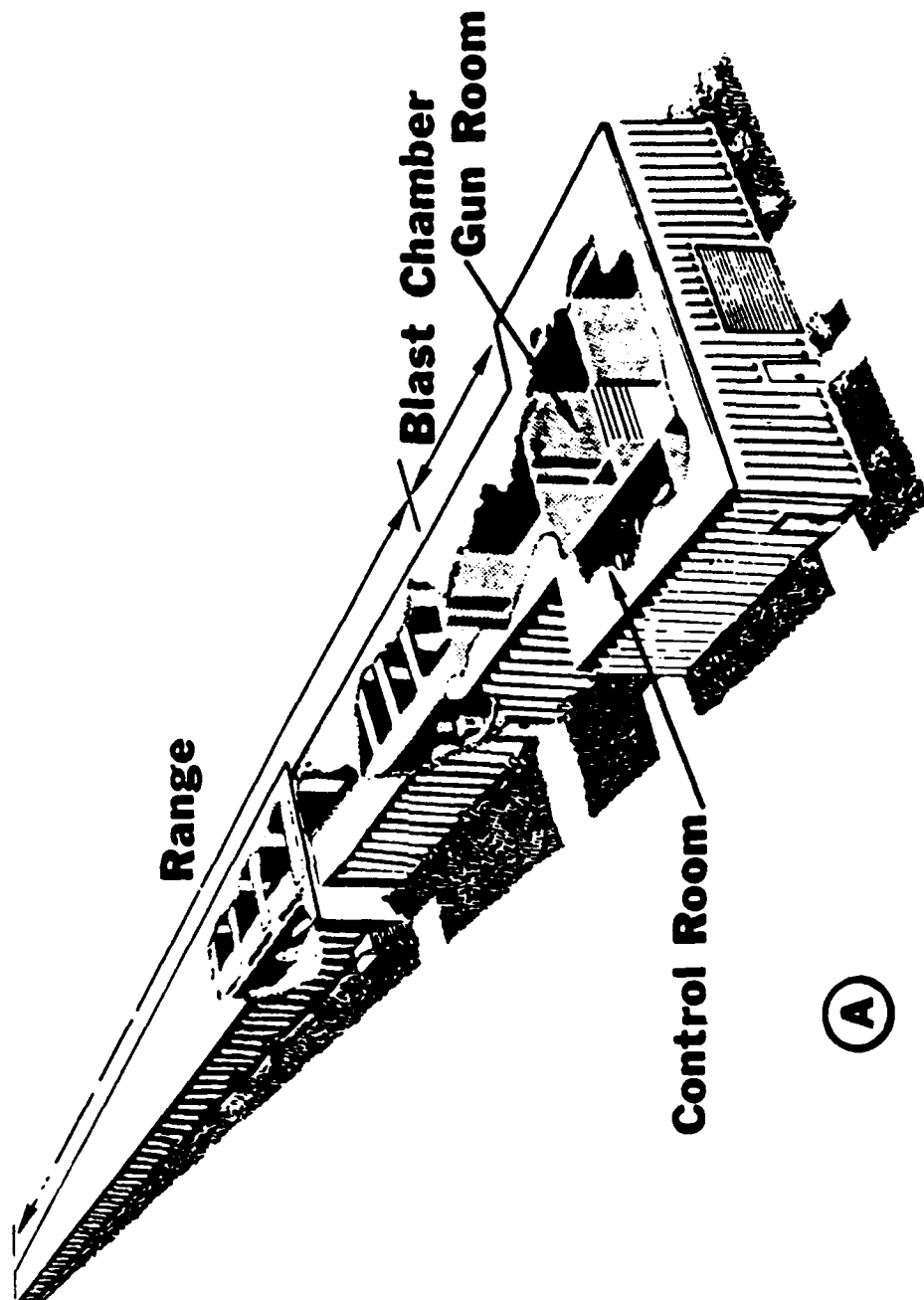


Figure 1. Aeroballistic Research Facility

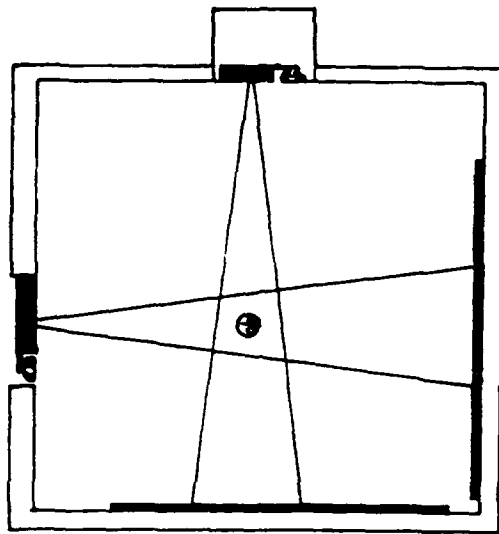


Figure 2

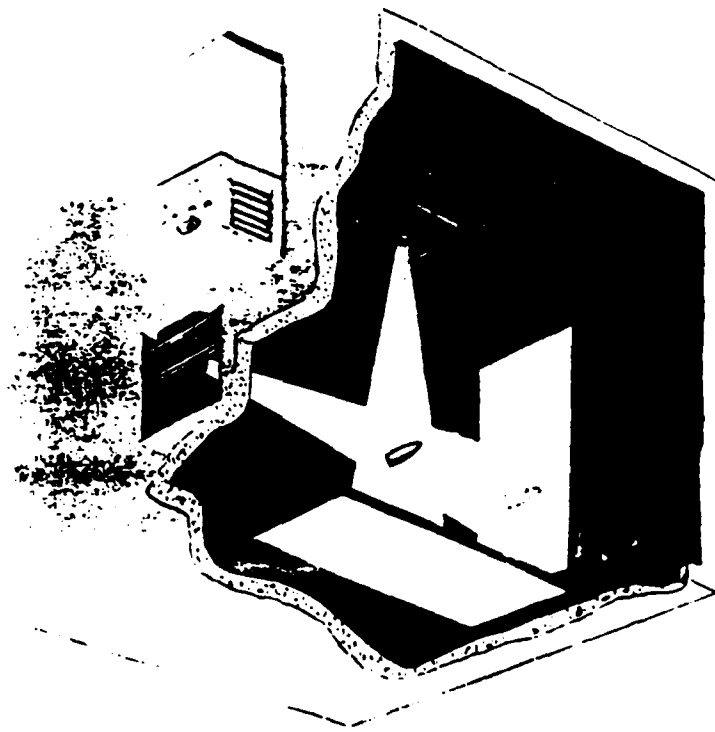
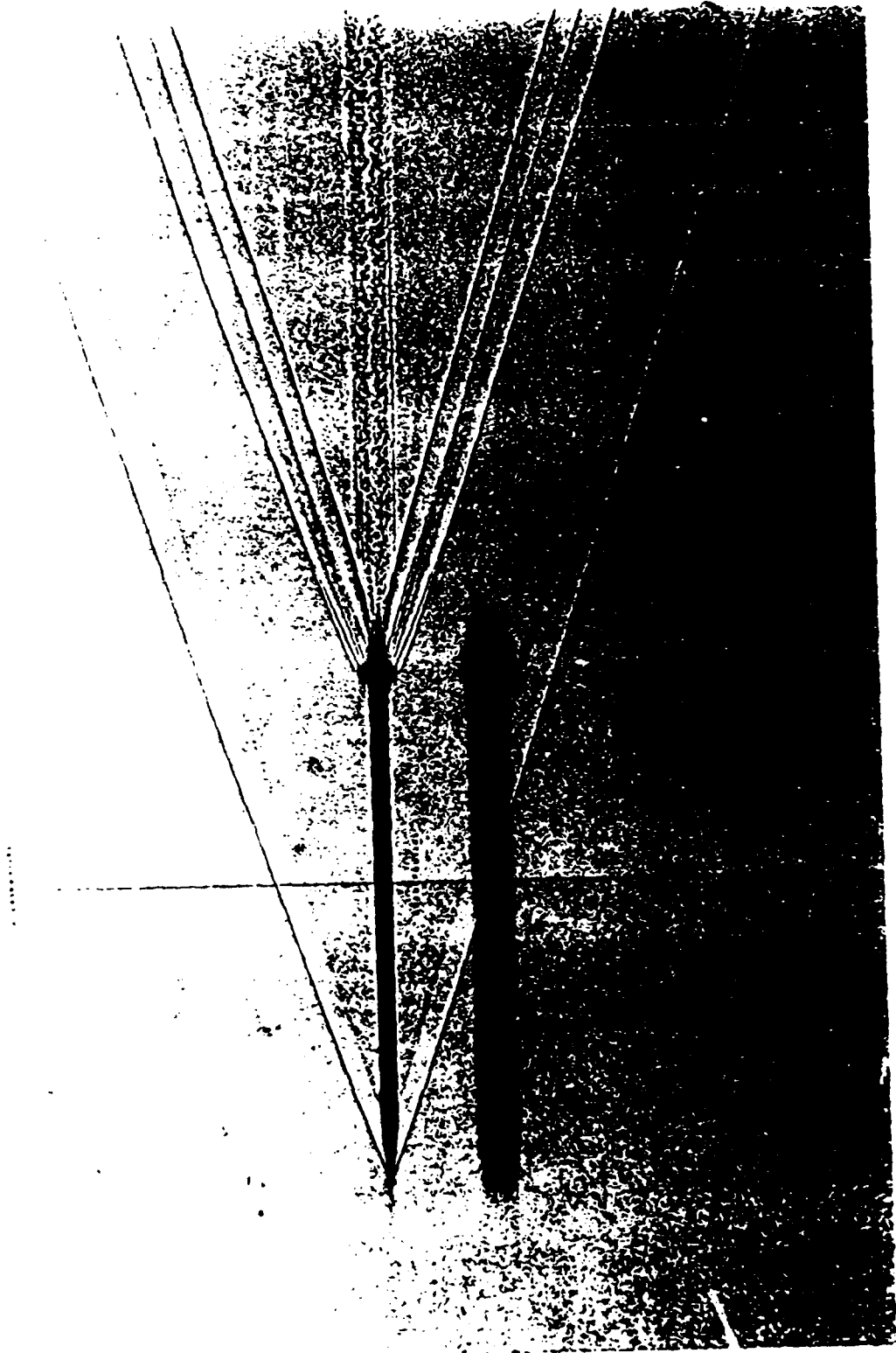
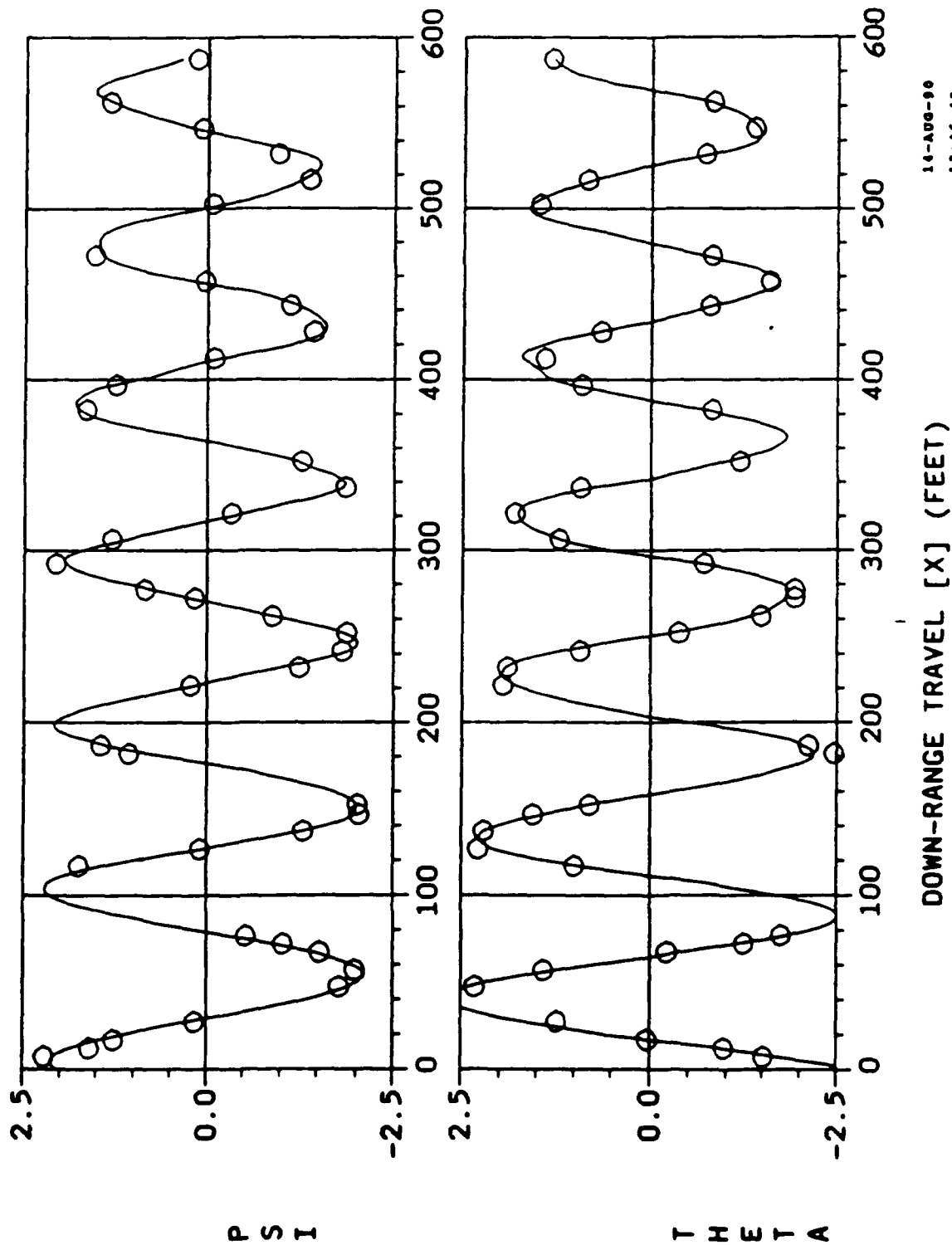


Figure 3



BS89061260 HVM-SBN 1



14-AUG-90
12:46:49

Figure 4

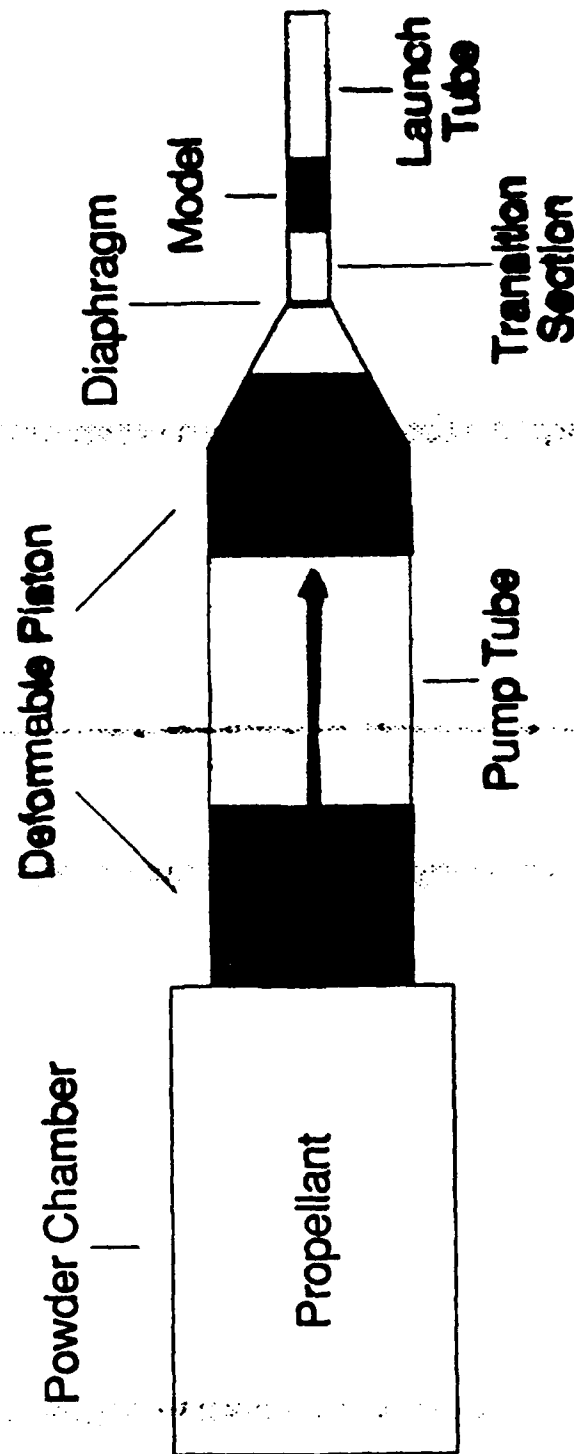
IV. Description of Research

I, along with the engineers, work in a small building near the ARF. We work primarily with exterior aerodynamics--the missile's flight properties between muzzle exit and impact. In order to obtain the desired initial condition for exterior ballistic test, a knowledge of interior ballistics is necessary, also. Interior ballistics work is concerned with the gun dynamics before the projectile exits the muzzle. The internal dynamics of the gun have a great affect on the projectile's motion, which includes velocity, spin rate, angle of attack, and break-up due to stresses on the projectile in-bore. My main project this year dealt with a program designed to simulate the interior ballistics of a light gas gun.

Figure 5 is a diagram of the light gas guns used by the ARF and BEF. The back section is a powder chamber. When the powder explodes, it forces the piston through the second section, a chamber filled with helium. The helium is forced toward the diaphragm and is compressed until the diaphragm, at some design pressure, bursts. The escaping gases then propel the model out of the launch tube, and the deformable piston stops in the transition section.

Several years ago, Dr. Bob Courter, a professor at Louisiana State University, and one of his graduate students created a relatively simple program to simulate the internal dynamics of the light gas gun. This program helped our branch to predict the muzzle velocities of projectiles at given parameter values. Captain Mike Valentino, my mentor, wanted to make graphs using the program showing the relationships among gun properties for different powder weights, projectile weights, and gun bore diameters. Such graphs would require too many iterations of the program to be practical, so Captain Valentino asked me to develop a more expeditious method for creating

TWO STAGE LIGHT GAS GUN



- LAUNCHES PROJ TO MACH 10+
- PRODUCES SOFTER LAUNCH
- TAILOR MADE FOR AEROBALLISTIC RESEARCH

Figure 5

these graphs.

The original program asks the user to enter seven parameters: projectile package weight, charge (powder) weight, piston weight, initial helium pressure, pressure at which the diaphragm fails, initial gauge pressure, and ambient temperature. After entering these values the program begins its calculations. When it finishes--after approximately one to three minutes if the entered values are correct--it asks if you desire a plot of the results. Unfortunately, this plotting routine does not work.

When Captain Valentino assigned me this task, I began work to make the program capable of iterations. The complexity of the mathematics mandated reinitializing every variable before each subsequent iteration. Consequently, the values for the parameters would have to be stored externally on a sequential file to keep them from being reinitialized, and realizing that, I created two separate programs. The interactive program inputs and stores the values for the two important parameters, projectile package weight and charge weight. I altered the main program to set the values of the other parameters to common standards. The input program itself is simple. It asks for the gun barrel diameter (the ARF and BEF use 20, 30, and 40mm Light Gas Guns). Next, it asks for the number of projectile weights and then the number of charge weights. Lastly, it asks for all the projectile weights and powder weights to be used and stores them in a sequential file as a two column matrix of up to 100 permutations. Attachment two is a copy of the input program and the modified source code.

I modified the source code to input the projectile weight and powder weight data from the file and use them to calculate predictions. I also modified it to include maximum acceleration. The program calculates, using integration, new values for velocity every .1 millisecond. As a result, it

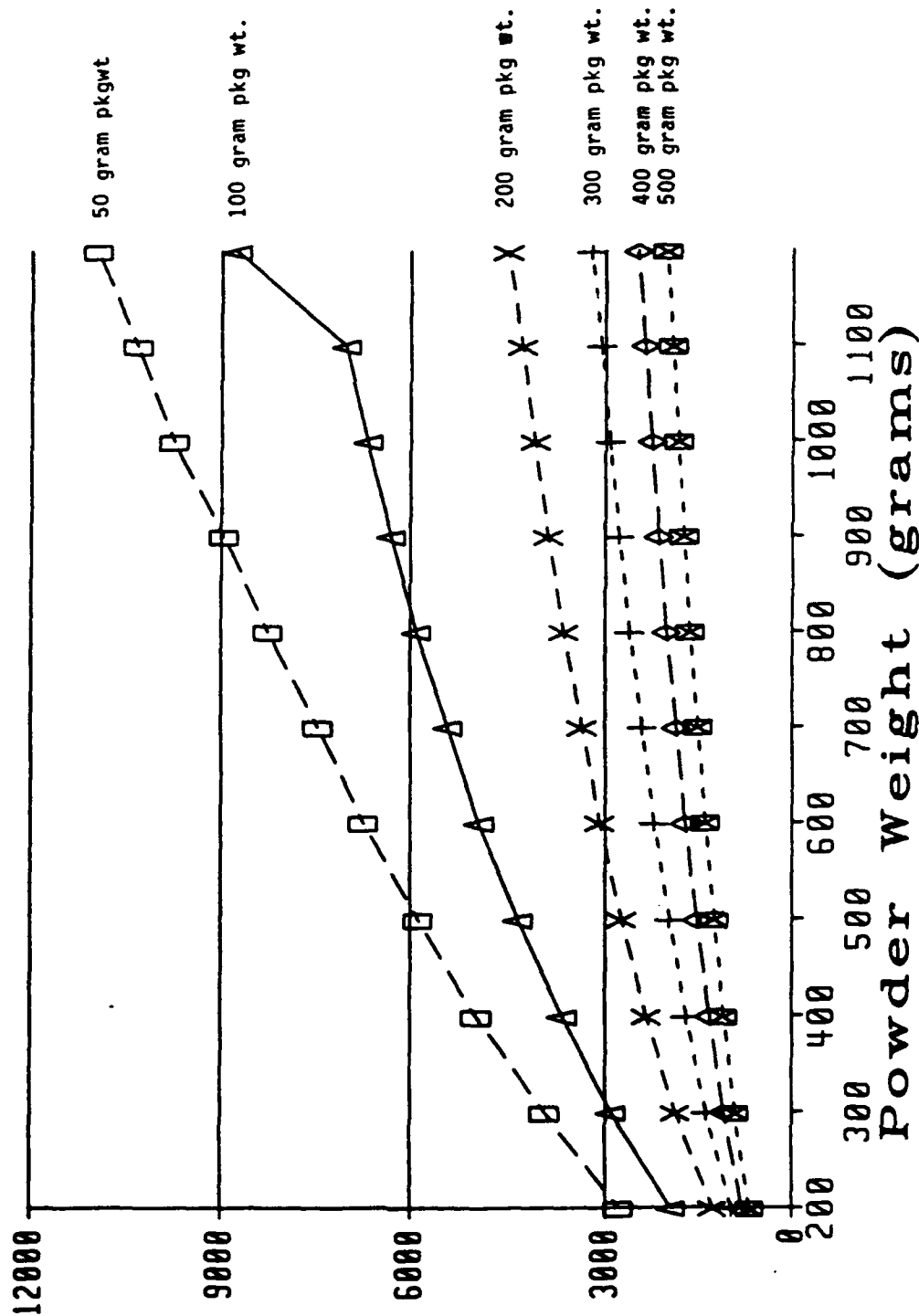
could be easily modified to calculate acceleration from the formula $(Vel1 - Vel2) / (Time1 - Time2)$. The values obtained for acceleration are important because the models can survive only a certain acceleration without breaking. If the predicted acceleration is greater than the model can withstand, the engineers know to adjust the physical parameters before shooting to ameliorate the loads on the model.

Because Captain Valentino had requested graphs of the output, an output file of important results had to be created. This output data included the following: package weight, charge weight, muzzle velocity, maximum acceleration, and maximum pressure. The plot output, is written to a file in ASCII format, a format readable by many spreadsheet graphics programs. The ultimate goal is to read the plot file into the spreadsheet and make graphs which might express more clearly any trends or relationships between powder weight and projectile weight, and reveal any latent inconsistencies in the predictions.

The results of this effort can be seen in figures 6-8. Manually entering the data in these charts would take days, but with the modified program, it takes just over an hour during which time the user may leave the computer. Once started, it runs until completion. A batch file calls the input program, values are entered, the batch file calls the main program, and the computer proceeds to calculate results for every "shot" until it is finished. The plot file can then be read by Enable, a spreadsheet program, and the graphs are made.

Figure 6

Performance of 20mm LGG (10lb Piston 400PSI He)



Performance of 20mm LGG (10lb Piston 400PSI He)

Figure 7

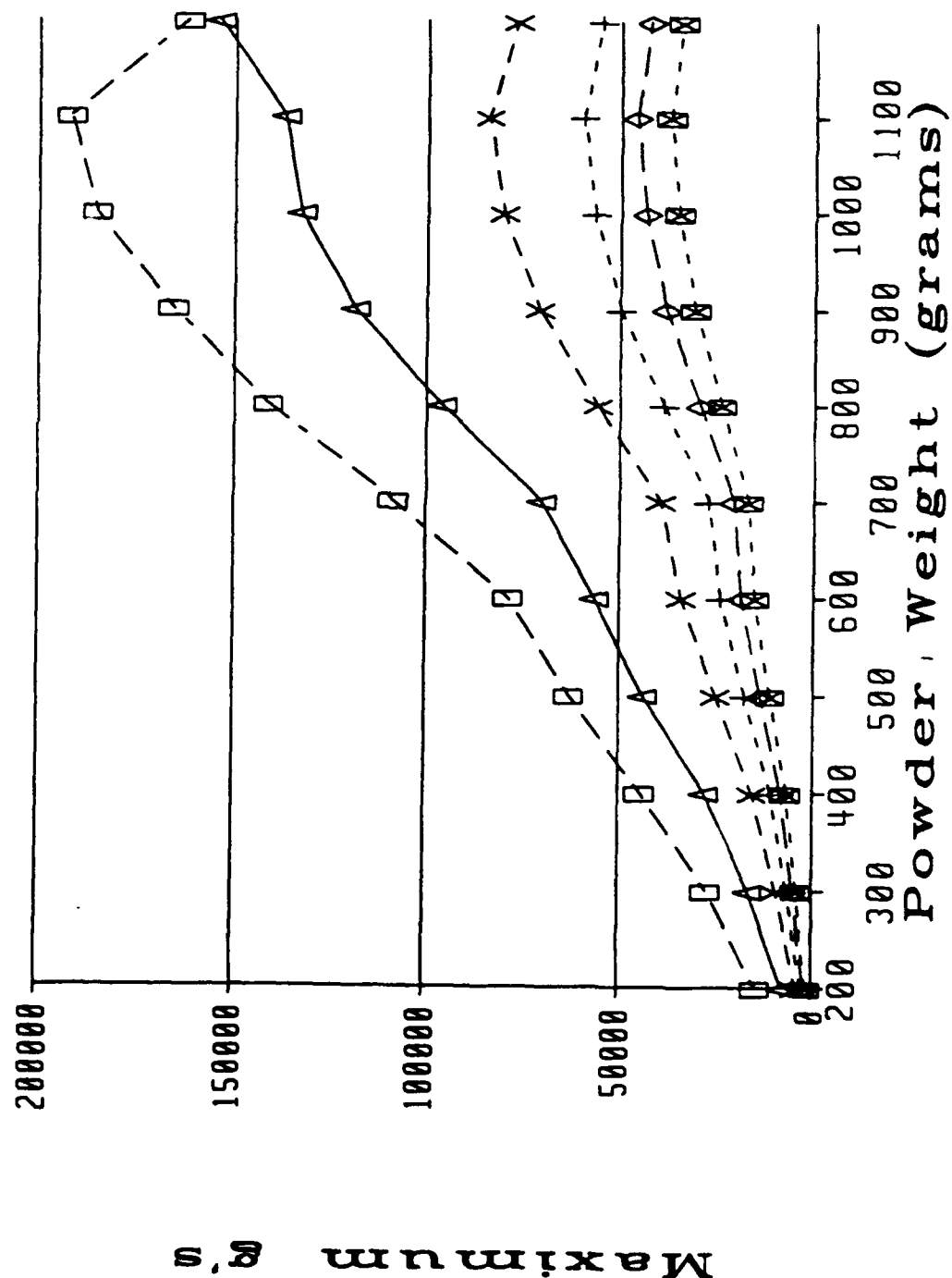
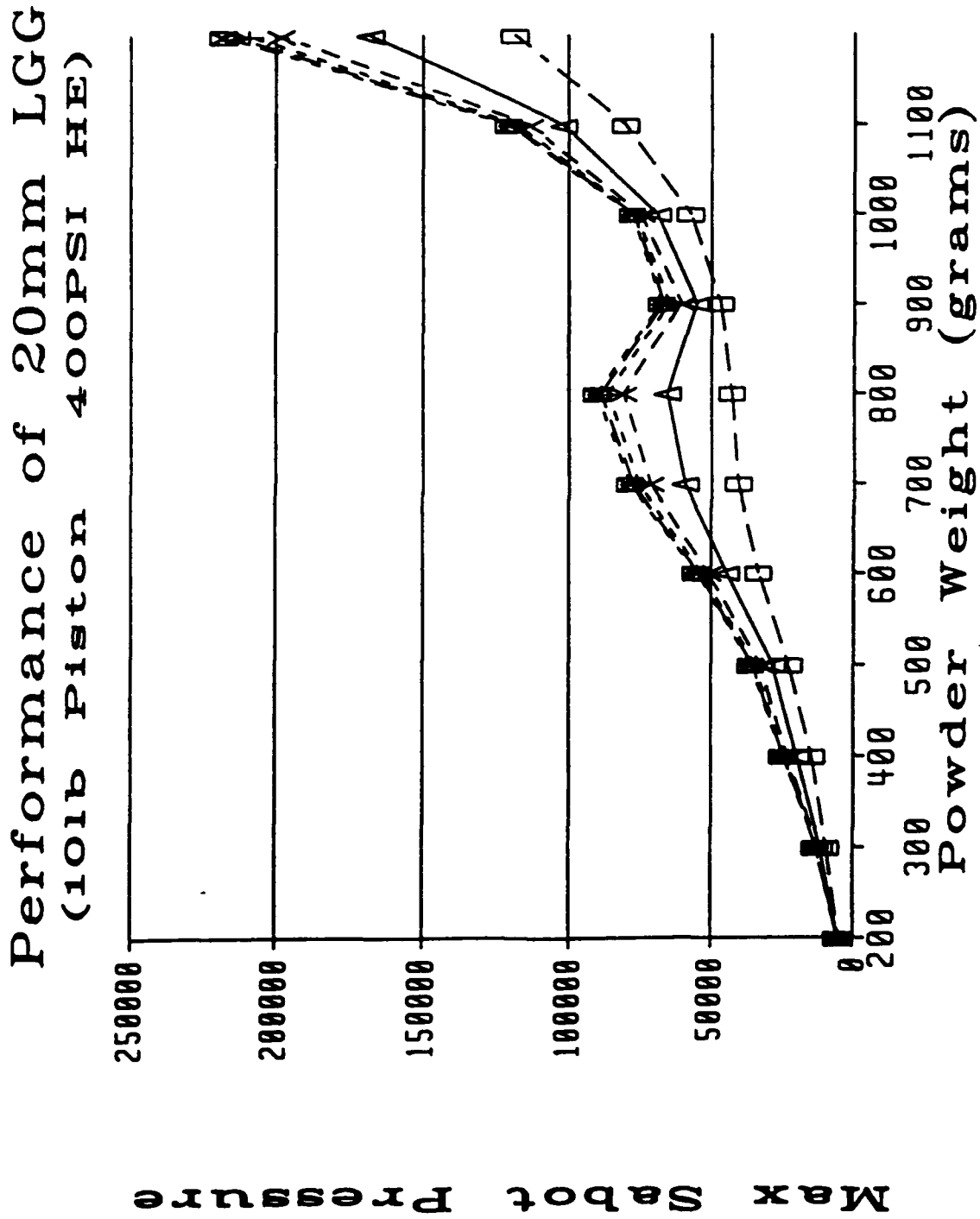


Figure 8



V. Results

I made some important discoveries during my work. The program reads a data file of various constants, but until my research, we did not know what the values in the file represented. We learned, for example, that the data file read by the program uses the properties of the powder CIL3331 for the simulation of the internal dynamics. A different powder could yield much different results. Contrary to our expectations, the twenty millimeter gun shot the same projectiles slower than the thirty millimeter; although, the thirty millimeter shoots faster than the forty millimeter. Also, the program seems to err in predicting certain high projectile weight, high powder weight values for the thirty millimeter gun.

VI. Conclusion

The new program will help the branch save time and money during testing. Because we know what the constants in the data file represent, it will be possible in the future to change the program to accommodate changes in the guns or powders. The ability to run multiple "shot" simulations quickly will help save time when trying achieve a specified velocity. Having the ability to see 100 shot predictions at one time on a graph may also help reveal any program bias. Most importantly, money will be saved because fewer models will be wasted during testing.

VII. Experiences Gained

During my brief employment at the lab this summer, I made similar graphs for our powder guns from a program called PRODAS. PRODAS is a ballistics simulation program on the VAX/VMS system. These graphs were relatively easy to make because no modifications to the program were necessary. I also used PRODAS to simulate the trajectories of a 50 caliber M33 in various wind conditions. This data will be used to help develop new targeting systems for guns in all branches of the military. Figures 9 and 10 show some cross plots of the results.

Although it has not been completed, I have tried, on my own time, to write a simple program to calculate a projectile's trajectory. This program draws on all aspects of my experiences these last two summers. First, the program is being written in FORTRAN, a language I was first introduced to last summer. I am continuing to improve my FORTRAN skills. The equations the program uses, because they include drag, are ones I learned here. The inspiration to write the program came from a personal desire to synthesize my experiences with simulation programs, my knowledge of FORTRAN, and the mathematical tools I learned on the job.

Figure 9

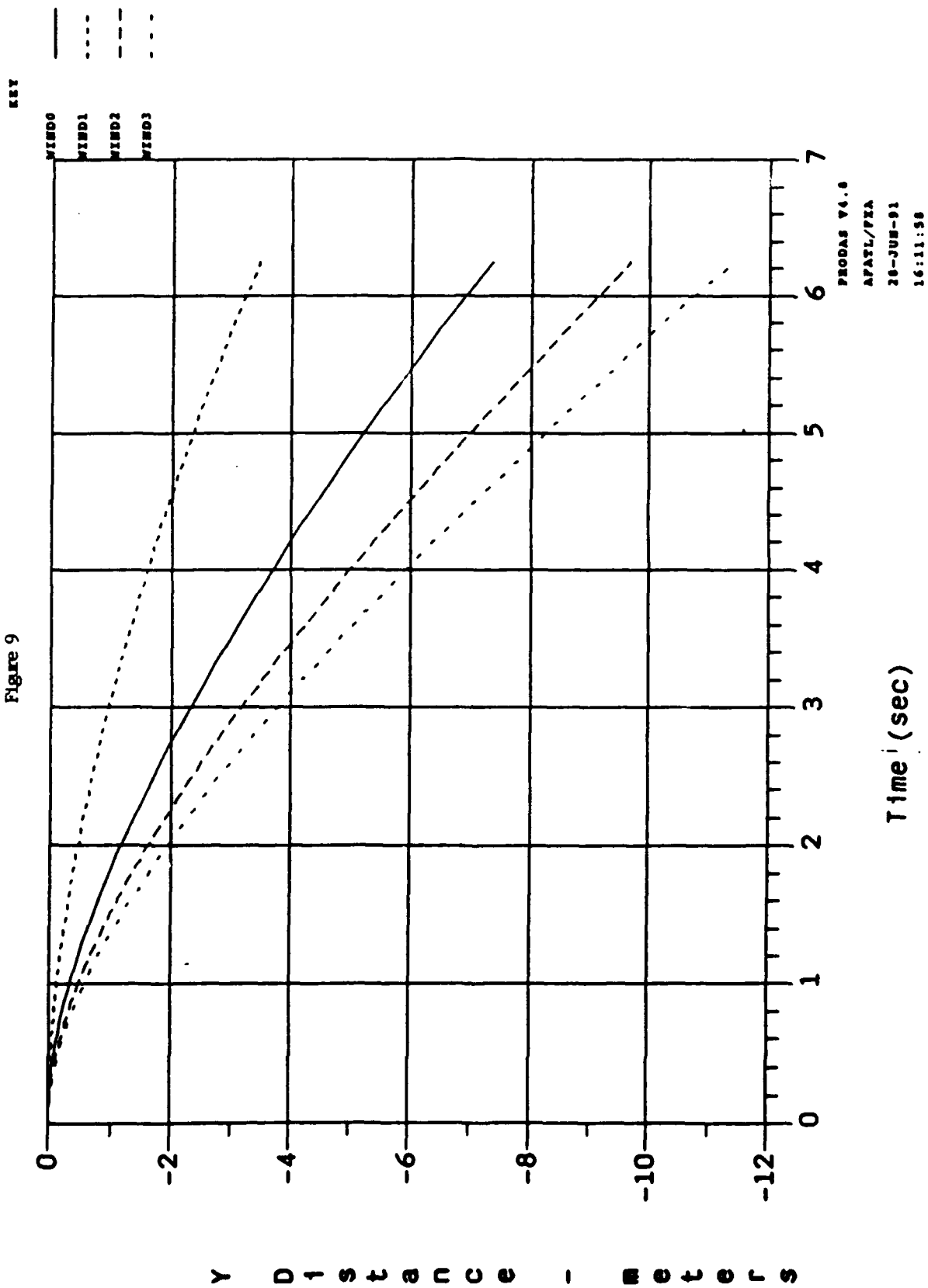
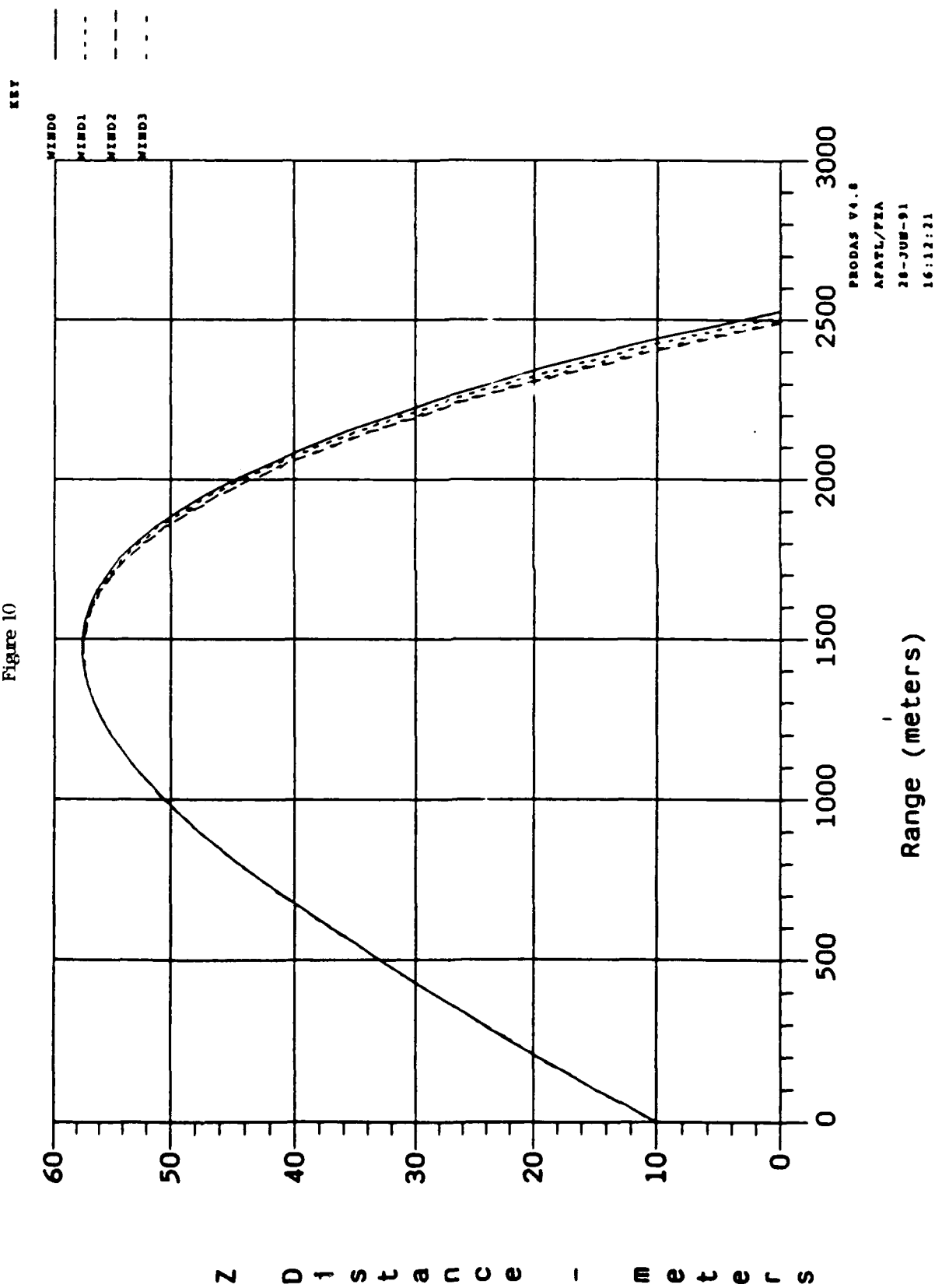


Figure 10



VIII. Bibliography

Anderson, John D., Jr. Introduction to Flight. 2nd Ed. McGraw-Hill: New York, 1985.

Ballistic-Range Technology. edited by Thomas N. Canning, et al. Technical Editing and Reproduction LTD: London, 1970.

Kittyle, Robert L., et al. Description and Capabilities of the Aeroballistic Research Facility. AFATL-TR-87-08, 1987.

(ATTACHMENTS ON FILE AT RDL.)

RANDOMLY GENERATED SYNTHETIC BACKGROUND DATA USING ITERATIVE FRACTAL TECHNIQUES

Eric P. White, WL/MNGA HSAP 1991

INTRODUCTION

Working in such an educational and constructive environment as the Eglin Armament Directorate of Wright Laboratories leads to mental growth beyond anything taught in any school. The research that takes place and the facilities that it uses are on the leading edge of technology. No other work experience could possibly rival that of the programs of the Air Force laboratories.

This is my second year to participate in the High School Apprenticeship Program. Last summer I researched the feasibility of using fractals to generate realistic background terrain for simulation purposes. I also wrote an example program under the Turbo C programming language on the IBM compatible personal computer in our office.

This year I worked in the Advanced Guidance Division under the same mentor as last summer, Mr. Lee Prestwood. My assignment was to move my project to the Image Processing Laboratory which my division supports. Here the program would become a utility available to any program supported by the IPL.

DISCUSSION

The IPL deals very often in the development and testing of target acquisition algorithms to find and identify possible targets amid background noise. We decided that a utility to produce this background noise would be very useful in many applications.

Last summer my project concluded that realistic background scenery could be generated quickly and easily. It also pointed out the advantages to using synthetic generated background data as opposed to creating images from actual photographs of either real scenery or scale models of scenery. Computer generation boasts speed and cost efficiency that other methods lack. This led to the research into a feasible algorithm to prove my point. The method I chose is known as midpoint displacement.

Midpoint displacement is an iterative fractal technique that deals with the fractal property of self-similarity. Basically this means repeating the same procedure on a smaller and smaller scale until the result meets the desired amount of detail. This same property of self-similarity has many examples in nature. Notice, for example, how a tree's branches resemble the original tree, or how a portion of coastline looks remarkably similar regardless the distance from which it was viewed (fig.1). Through mimicking nature we are able to achieve realistic models of background terrain.

Midpoint displacement begins with a simple geometric shape, a square (fig.2). It then calculates the midpoints of

the sides of the square and modifies the values of these midpoints. The new value is calculated based on the average of the endpoint values of the segment used and a random modifier added to that. Next, the center of the square is found and modified based on the average of the corner values of the square and a random modifier added to this value. This square is then divided into four new squares and the process repeats itself. The resolution becomes more detailed as the number of squares increases.

The first program I used in my project was adapted from Amiga Basic to Turbo C for use on an IBM compatible personal computer. The Turbo C language provided many useful graphics routines that made plotting the data received from the program very easy. This Turbo C version was the basis of my project last summer.

The next version came this summer and was written in VAX C for use with the DECwindows environment. After the main portion of the code had been transferred to VAX C on the IPL's VAX 8650 mainframe computer, the DEC Xlibrary routines provided the windows interface. The syntax for writing DECwindows programs proved very challenging to learn, and the program was not completely successful. Since the program was merely a utility, it had little usefulness in such a complex environment, so I deleted the windowing routines. Afterwards, for efficiency purposes, I streamlined the program and removed

the plot routines. This proved more beneficial to meeting my goal.

RESULTS

My efforts succeeded in producing a very efficient VAX C program to be used as a utility on the Image Processing Lab's VAX 8650 mainframe. The program asks for several simple parameters to run. It has user-definable resolution, it scales the values to within desired specifications, and it allows the user to input the variation between values. Average run time of the utility is between thirty and sixty seconds depending on resolution and CPU availability. The program writes the data for the elevation values in a file of two byte integers that may be easily displayed on the Gould IP8500 Image Processing and Display System of the IPL. Any users of the IPL computer resources may have free access to this utility to use as they please.

CONCLUSION

My project has proven to be very efficient and useful. The midpoint displacement algorithm serves its purpose very well. The summer has ended with success, but with many learning experiences on the way. As well as writing software, I assisted with maintenance in the IPL. My mentor, Lee Prestwood and I were responsible for reconnecting the Gould IP8500 Display System with new, space saving coaxial cables

that provide a better connection throughout the system. Also, we assembled and installed monitor stands in the lab that cleared desk space for more users. Overall I learned more with actual work experience in the lab than any school class could ever hope to teach.

ACKNOWLEDGEMENTS

I'd like to think everyone in the Image Processing Lab and Radar Signal Processing Lab for helping me learn the system and answering my many questions. This includes Larry Neal, Dave Kerr, Noreen Porter in the IPL, and Steve Halprin and Pat Carlton in the RSPL next door. I'd especially like to think Mr. Lee Prestwood for providing me with the opportunity to work in the lab and more support than I could have ever expected.

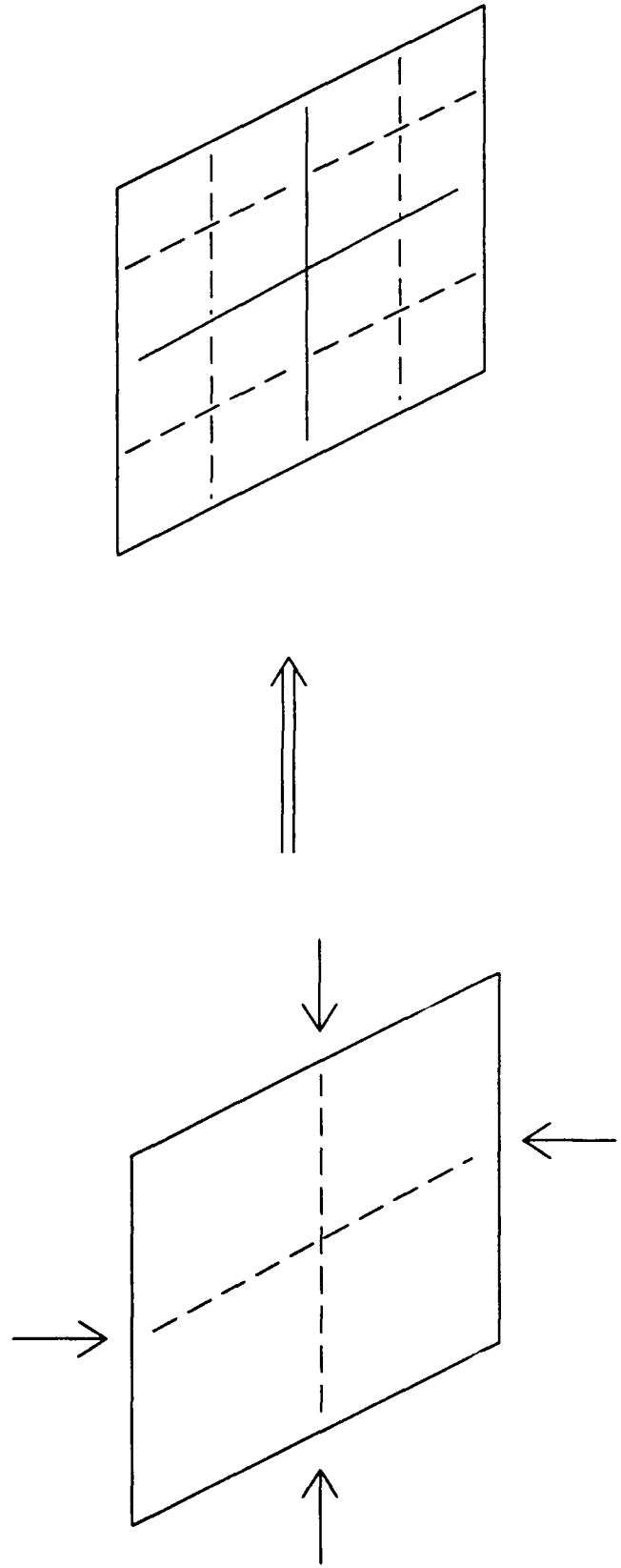
The Fractal property of Self-Similarity



General Repetition of Shapes in Nature

The Concept of Midpoint Displacement

- Start with a geometric figure such as a square
- Calculate the midpoint of each segment
- Modify the value of these points a random amount
- Repeat with newly formed figures




```

/*****      Fractal landscapes      *****/
/****
/****      Generates random background data for      ****/
/****      use as noise in testing situations      ****/
/****-----****/
/****      Programming by Eric P. White, HSAP 1991      ****/
/****-----****/

#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <file.h>
#include <math.h>
#include <time.h>

/**** Global Variables ****/
double elevation[513][513], max_elevation = 1, min_elevation = 1;
int new_seed, minvalue, maxvalue;

/**** Function prototypes ****/
int scale (double, float, float);
void make_mountain (float, int);

/**** Main program segment      ****/
main ()
{
    int num=0, num2=0;
    int resolution=512;
    int size;
    int exponent;
    unsigned short scaled_elevation;

    float percent=110, maxvar=2.0;
    FILE *outfile;
    unsigned short *buffer;
    char filename[81];

/**** User input block ****/
    printf("\nEnter filename to output data: ");
    gets(filename);

    printf("\nEnter the resolution of the plot (max=512) ");
    scanf("%d", &resolution);

/**** allocate area of memory for pointer ****/
/**** acts as a variable length array ****/
    buffer = (short *) calloc(resolution,sizeof(short));

    do
    {
        printf("\nEnter the maximum variation for the elevation (0-2) ");
        scanf("%f", &maxvar);
    } while ((maxvar < 0.0) || (maxvar > 2.0));

    do
    {
        printf("\nEnter percent of flatland/water ");
        scanf(" %f", &percent);
    } while ((percent < 0.0) || (percent > 100.0));

```

```

do
{
    printf("\nEnter max elevation value. ");
    scanf(" %d", &maxvalue);
    printf("\nEnter min elevation value. ");
    scanf(" %d", &minvalue);
} while ((minvalue < 0) || (maxvalue > 255));

printf("\nEnter a random seed ('1' for default starting point) ");
scanf(" %d", &new_seed);

/**** End user input block ****/

/**** initialize array ****/
for (num = 0; num <= resolution; num++)
    for (num2 = 0; num2 <= resolution; num2++) elevation[num][num2] = 0;

/**** call function to fill elevation data array ****/
make_mountain (maxvar, resolution);

/**** Open file to write data in binary format ****/
if ((outfile = fopen(filename, "wb")) == -1)
{
    printf("!Error opening file!");
    exit(0);
}

/**** Write elevation array to data file in 2 byte integers ****/
for (num = 0; num < resolution; num++)
{
    for (num2 = 0; num2 < resolution; num2++)
    {
        *(buffer+num2) = scale(elevation[num][num2], maxvar, percent);
    }
    fwrite(buffer, 2, resolution, outfile);
}

}
/**** end main program segment ****/

/**** Function to fill elevation array ****/
/**** ****/

void make_mountain (float max, int resolution)

/**** routine loads randomized values into an array
based on the neighboring values ****/

{ int iter, side, x, y;

double old, ran, random_value;

int half_length;

int exponent;

div_t result;

old = 0; iter=0; side=0; x=0; y=0;

```

```

x = 0;
srand(new_seed);
frexp(resolution, &exponent); /** Iter must be log base 2 of ***/
                               /** the resolution.      */

/**** main loop ****/
for (iter = exponent; iter > 0; iter--){

    side = pow(2, iter); /* side is defined as 2 to the power of the
                           current iteration */

    result = div (side, 2);
    half_length = result.quot;

    printf ("\nDoing iteration %d \n", iter);
    printf ("Tops & Bottoms \n");

    /**** find values for the tops and bottoms of the squares ****/

    x=0; y=0;
    for (y = 0; y < resolution; y += side)
    { /* loop for y value of array */
        for (x = half_length; x < resolution; x += side)
        { /* loop for x value */
            ran = (rand()/(2147483647.0)); /* calculates the */
            random_value = ran * max * side; /* random value */
            old = (elevation[x-half_length][y]) +
                (elevation[x+half_length][y]);
            old = old/2; /* allows new value to be proportional to old */
            elevation[x][y] = old + random_value; /* sets new elevation */
        }
    }

    /* find values for the sides of the squares */
    printf ("Sides \n");
    x=0; y=0;
    for (x = 0; x < resolution; x += side)
    { /* loop for y value of array */
        for (y = half_length; y < resolution; y += side)
        { /* loop for x value */
            ran = (rand()/2147483647.0); /* calculates the */
            random_value = ran * max * side; /* random value */
            old = (elevation[x][y-half_length]) +
                (elevation[x][y+half_length]);
            old = old/2; /* allows new value to be proportional to old */
            elevation[x][y] = old + random_value; /* sets new elevation */
        }
    }

    /* find the values for the centers of the squares */
    printf ("Centers \n");
    x=0; y=0;
    for (x = half_length; x < resolution; x += side)
    { /* loop for x value of array */
        for (y = half_length; y < resolution; y += side)
        { /* loop for y value */
            ran = (rand()/(2147483647.0)); /* calculates the */
            random_value = ran * max * side; /* random value */
            old = (elevation[x-half_length][y+half_length]) +
                (elevation[x+half_length][y-half_length]) +
                (elevation[x+half_length][y+half_length]) +
                (elevation[x-half_length][y-half_length]);
            old = old/4; /* averages 4 values instead of 2 */
        }
    }
}

```

```

        elevation[x][y] = old + random_value; /* sets new elevation */
        if (elevation[x][y] > max_elevation)
            max_elevation = elevation[x][y];
        if (elevation[x][y] < min_elevation)
            min_elevation = elevation [x][y];
    }
}
/**** end main loop ****/

}
/**** end function make_mountain ****/

/**** Function to scale elevation array value ****/
/**** ****/

int scale (double value, float maxvar, float percent)

/**** routine scales a value to a usable range ****/

double new_elevation;
{
    float flatline, distance;

    /**** scales value to between 0 and 255 initially ****/
    /**** and levels value below flatline *****/

    flatline = fabs((max_elevation-min_elevation)*percent/100);
    value = value-flatline;
    if (value < 0) value = 0;
    value = (value)/ (max_elevation-flatline);
    new_elevation = value*127.5*maxvar;

    /**** scales value a second time *****/
    /**** to between user definable values ****/

    distance = 255.0 - ((255.0-maxvalue) + minvalue);
    new_elevation = (distance/255.0)*new_elevation + minvalue;

    /**** returns the new scaled elevation ****/
    return (new_elevation);
}
/**** end function scale ****/

```

ABSTRACT

Background noise, namely terrain data, for use with various detection routines would be very useful in a computer laboratory. The lab in which I work provides support for projects that very often deal with such detection routines. It was decided that a utility to generate random background data would be an asset to the Image Processing Laboratory. Numerous methods to do this were researched to determine the most efficient and useful. The final product was a simple fractal based utility written in the C programming language that wrote the random generated data into a file for further manipulation.

BIBLIOGRAPHY

Barkakati. The Waite Group's Turbo C Bible. Howard W. Sams and Company, Indianapolis. 1989.

Peitgen, Heinz-Otto and Saupe Dietmar. The Science of Fractal Images. Springer-Verlag, New York. 1988.

Timmerman, Matthew. "Fractal Mountains for Amiga". Compute!. Aug., 1987. pg.89-91.

SOFTWARE DEVELOPMENT FOR THE AEROSOL TEST CHAMBER
AND A CODE CONVERSION FOR A DATA RETRIEVAL SYSTEM

James E. Youngblood

Mike Caraway--Mentor

August 14, 1991

INTRODUCTION

During this summer I had an opportunity to work on two projects. The first, SOFTWARE DEVELOPMENT FOR THE AEROSOL TEST CHAMBER, took up the majority of my time. I was told that I would write some software in C for the embedded system that will controll the chamber. I didn't even know what an embedded system was! I eventually learned a lot about the Aerosol Project and how much time and effort goes in to a project like this.

My second project, PASCAL TO C CODE CONVERSION FOR A DATA RETRIEVAL SYSTEM, was given to me a few weeks before I would leave in order to present me with some idea of what program maintenance is all about and why commenting your code is important.

SOFTWARE DEVELOPMENT FOR THE AEROSOL TEST CHAMBER

Fuze sensors are currently tested against real-world conditions by using filters or mathematical equations. The filters work well when just testing the power of light that would be reflected by the target through the aerosol. However, they do not consider the light that is reflected off of the aerosol particles back to the sensor. The mathematical testing does a little better but still can not emulate the real-world situations that occur in the job of a fuze.

The solution to testing the optical fuze sensors would be to create a real aerosol inside of a controlled environment. We are creating an aerosol test chamber (see Appendix A for diagram) which not only does that but does it automatically. It will be unique (no one else has one similar). It will be completely computer controlled and will have the ability to run multiple tests. All the user will have to do is tell the computer what shape of aerosol he/she desires. The embedded 8051 microprocessor will do the rest. A diesel pump will be used spray fuel on the heater to create the aerosol cloud. The cloud will then be moved down a 50 foot sewer pipe by a fan, also controlled by the microprocessor. There are fifty sensors (one foot apart) running down the tube to test the aerosol density at that location (to ensure that the aerosol profile in the tube is the same as the one

the user desired for the test). The fuze sensor will be placed at the end of the tube, and a target will be put close to the front. We can then simulate a cloud moving at a very high relative speed by slowing down the timing of the sensor. The aerosol's profiles will be displayed on a video terminal real-time as well as recorded on disk for later review.

My area of concern in the project was in developing the software (in C) that controls the embedded system. I was first given an overlook of the top-down structure and attempted to write the first few levels of the embedded software (see Appendix B). Because we are using an embedded system, we are severely limited in some areas. For example, we have no operating system to work with. We are having to write our own hardware drivers for everything. We have a 64K limit for code and data. Also because of the embedded system the compiler did not include many higher level functions such as video operations. (We can access via serial port a dumb video terminal.)

My tasks this summer under this project would include:

1. The algorithm design for the error correction of the motion and thickness of the aerosol including pattern matching capabilities (This would insure that the aerosol profiles would be accurate.).

And Coding for:

2. Serial initialization,
3. Video display control (including the user interface),
4. Controlling timers/counters for time critical actions.

1. I decided that for the purpose of algorithm design the data from the sensors would be put in an array, which I would call Actual (what the sensors are actually reading). The calculated data of what the aerosol should be reading at each location would be in an array called Expected. Therefore, if all goes well, the aerosol density stored in Actual should be equal to the aerosol density stored in Expected at all fifty of the sensor locations. We had to allow for problems that could occur, though. Problems that we knew would come about, especially since this system has never been operated before (so we don't know exact settings for the equipment) and no one else has a similar aerosol chamber for a reference.

I thought about possible things that could go wrong in the aerosol chamber:

First the pump may not be putting out the right amount of fuel to the heater. This could be the result of dust or dirt in the nozzle (or it just doesn't pump

like it should). This would result in an aerosol that was lower everywhere or higher everywhere (in density) than the expected one (see Appendix C). I wrote a routine that would check the overall ratio of Expected and Actual sensor readings and adjust the pump accordingly.

Second, I thought about if the fan were going at an incorrect rate. This would cause the aerosol to be crunched or spread out in the tube (see Appendix D). I wrote an algorithm that would check for certain point on the aerosol that it can locate in both the Expected and Actual sensor readings and adjust the fan speed accordingly.

The final possibility that I thought about could occur after a corrective action has been taken in either pump control or fan control. It would be a correctly shaped aerosol in the tube but would be incorrectly placed in the array that holds the data (see Appendix E). This would mean that, for instance, Actual (1) could be equal to Expected (2) and Actual(2) = Expected (3)...(The aerosol may be one sensor down from where it should be. The aerosol would be correctly shaped, however, and neither the fan or pump speed should be adjusted.) I wrote an algorithm that would check the sensors and shift the array if this was the case. I used a multi-level rating system to determine

the best match based on the closeness to a correct match and the distance from Expected that the Actual is.

2. To start off actually programming some code I used an IBM-compatible using an EMUL-51 card to interface the 8051 chip which was connected to a video terminal (VT-100). After writing any code I could compile it, load it into the emulator, and run it on the 8051 through the video terminal. The first program I attempted was to simply output a line of text to the terminal. This proved a little more difficult than it may seem. I first had to write some serial initialization functions. I later made these into an include file called 'serial.h' (see Appendix F). I then proceeded with the video functions.

3. For the video display I created several screen functions and wrote them into an include file called 'graphics.h' (see Appendix G). These included many of the familiar higher level functions included with most compilers made for microprocessors that have operating systems. I included functions that would move the cursor to different locations on the screen, change the color of the text, clear the screen, reset the terminal back into the desired text mode, and change the size of the text displayed.

I used many of these video functions when I started to write some of the user-interface and graph-displaying func-

tions. I began with gathering the information about how the user wanted the aerosol to be shaped. This was done by asking for the number of segments, the length of each segment and to what position each rose (in terms of aerosol density). I then converted this data into an array and displayed this information on screen in the form of a graph, and the user was prompted to see if the information was correct.

I was limited in how accurate I could display the graph because there was not a graphics mode to use. I had to display the information in the form of a bar graph created with text characters such as a solid box. These characters are 10 lines by 10 lines (100 pixels) which is a lot bigger and harder to work with than a graphics mode in which you would be able to put 1 to 4 pixels on the screen separately. This made it difficult to display the graph with very much accuracy. I set out to find a solution to this problem. What I ended up doing was using different characters in the graphics character set (ones that were 0 ,2 ,4 ,6, 8, or 10 lines high), instead of just the big block that was 10 lines high. This proved more accurate in the vertical bars of the graph. I still have not discovered a way and have decided that horizontal accuracy just doesn't go with text mode.

Later I turned the graphing instruction into a function so that it could be used to display the graph of any array passed to it. This would be useful because we now had a

function to display the current aerosol position to the user. I ran into another problem of the text mode here. We were going to be doing real-time tests. We were also only going to be able to use 2400 baud to communicate with the video terminal. This meant that using my bar graph to display the aerosol information would take a few seconds to update the screen just once. I needed a way to speed up this graph portion of the program. The solution I found was to graph just the top piece of the bars on the graph (not filling in the blocks below the top). I used a line located at 0, 2, 4, 6, 8, or 10 lines (pixels) high instead of a block that would fill everything below that line. The user could still tell the shape of the graph with a relatively good amount of accuracy and the function was much faster (see Appendix H). (To date we are now even considering getting a faster video terminal that will support 9600 baud.)

4. To gain knowledge of how to access and use the timers in the 8051 chip, I wrote a few functions that would setup up the Programmable Counter Array to keep time in seconds and a few other various things.

Code Conversion Project

A second, smaller project in which I participated was the Pascal to C conversion of the software which retrieves data from the Data Recorders in the testing of hard target fuzes. This testing includes firing into different materials, such as concrete, to test what an accelerometer registers while traveling through different materials. This data will be used to program the fuze with the ability to determine what material it is penetrating. For instance, if the accelerometer first registers the pattern for concrete and then dirt and concrete again, the computer inside the weapon would determine that it is penetrating a bunker and will know the most effective time to detonate.

The software that was currently being used to retrieve the accelerometer data from the recorders was written in Pascal. Since the person responsible for writing this software has moved to a different location and no one else knows how to program in Pascal, maintenance on the program was impossible. Inadequate commenting was also a problem. The solution was to convert the program into C, a language which many people in the branch know.

The conversion began slowly because I was not familiar with Pascal and was having to look most of it up in a reference book. I then discovered on one of my browses through a local Bulletin Board System, a Public Domain program put out

by Microsoft that would convert Turbo Pascal into Quick C. The converter worked well, and I was able to finish most of the code conversion (minus the commenting) during the span of my summer.

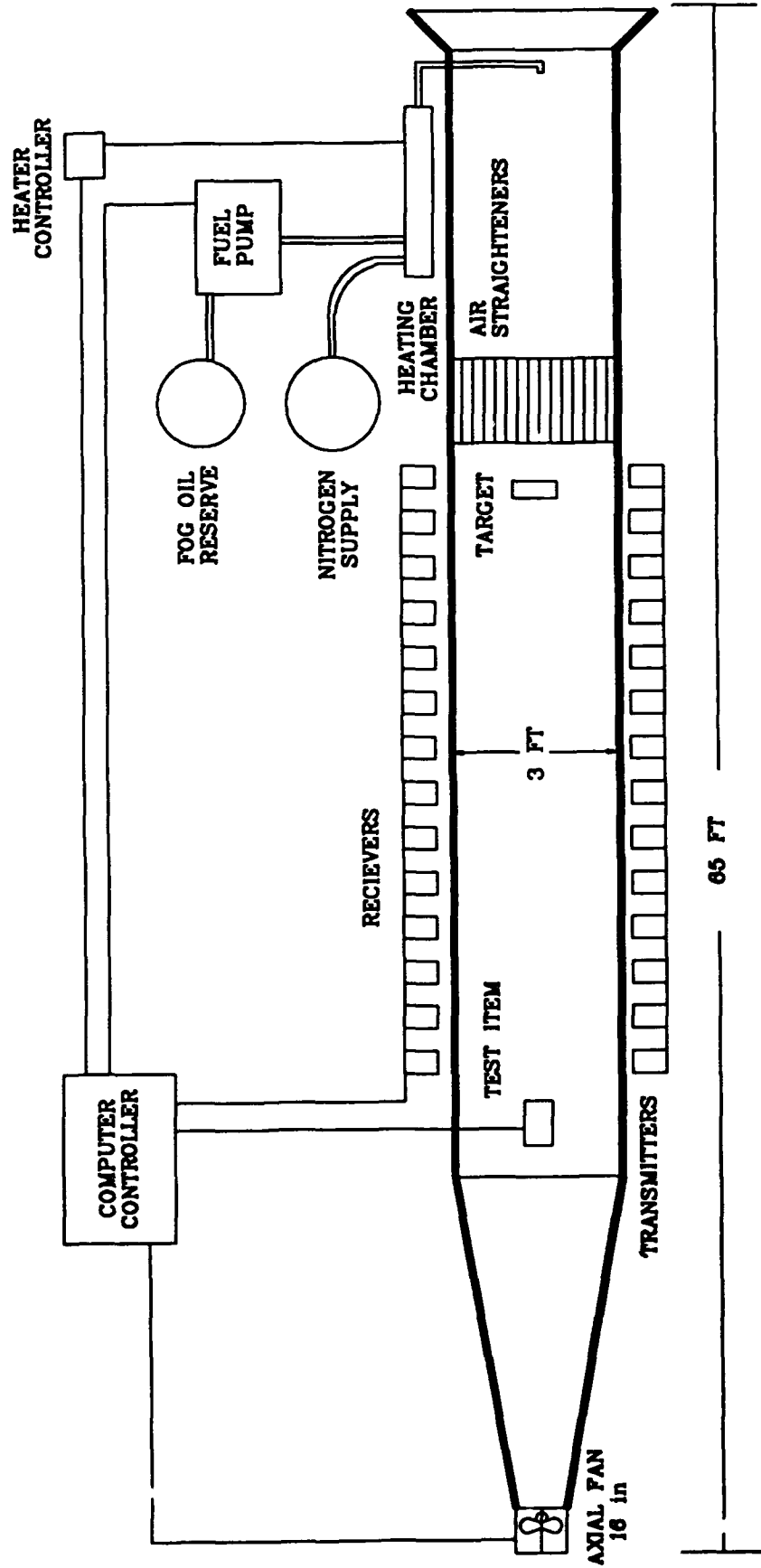
CONCLUSION

The Aerosol Project is still going together. Much of the software still needs to be written, and a lot of the hardware has not yet been built. I now realize how much has to be done in order to complete a project of this magnitude. I especially learned about the software end of it. Embedded programming is not something to be underestimated in terms of time taken and difficulty of programming or smoothness of operation and cost-effectiveness. This project gave me an opportunity to work with people who specialize in software and hardware, and made me realize how closely the two work together. (Also, I appreciate my operating system much more.)

The code conversion project has helped me get a better grasp of what maintaining a program entails (sometimes even translating the code into a different language), and I learned a little Pascal in the process. I also value the documentation and commenting of the code more (although I still don't like to do it).

APPENDICES

AEROSOL WIND TUNNEL



Aerosol Test Chamber:

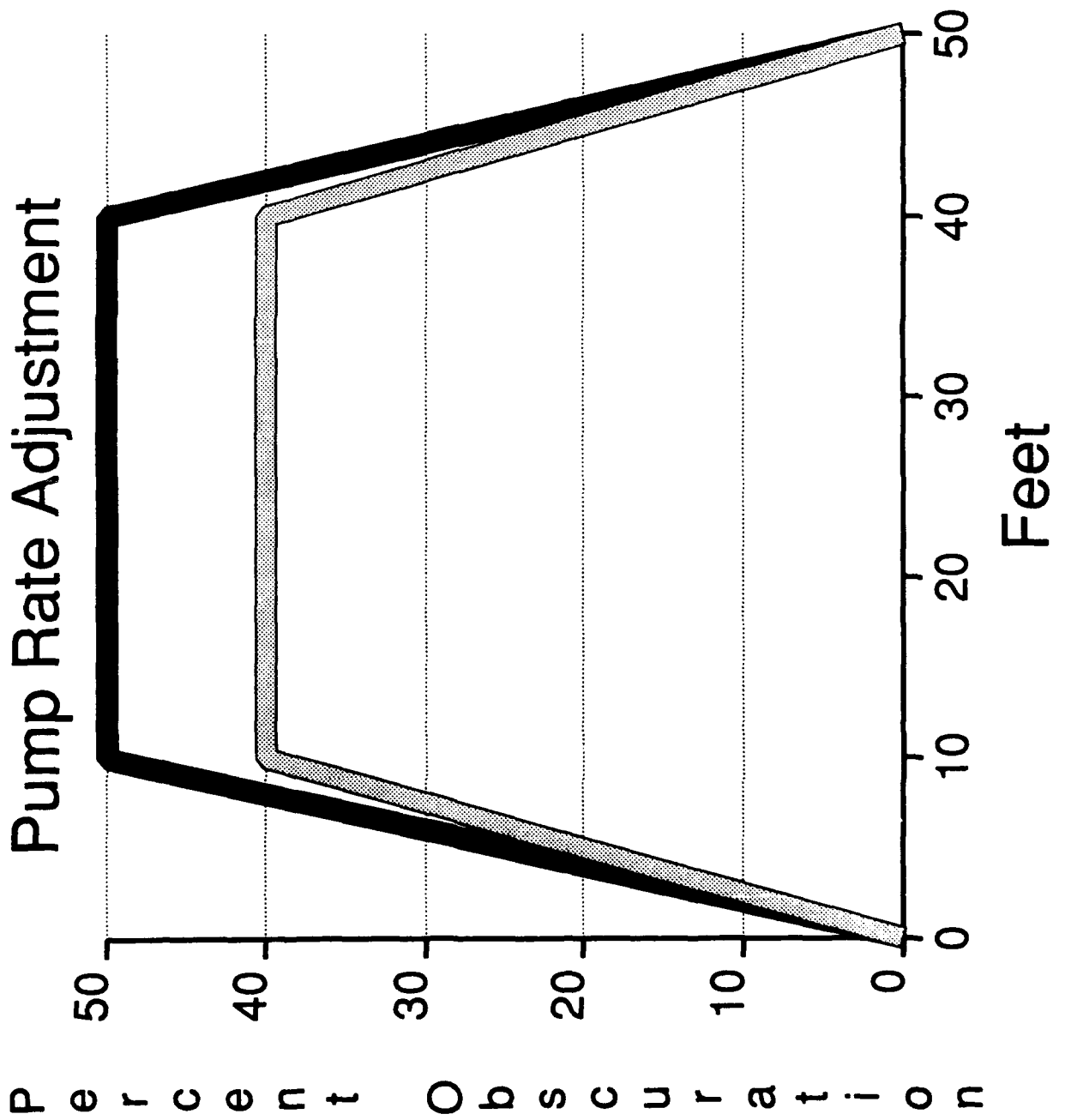
General Program Structure

<u>Initialize and Test Equipment</u>	<u>Actual Program</u>	<u>Shut Down Equipment</u>
Initialize and test fan	Check for emergency shut down	Turn off equipment
Init. and test heater	Ask user for data	Aerate room
Init. and test pump	Send data to equipment	Return to Main Menu
Report any errors	Read sensors	
	Display sensor data	
	Loop until test is finished or emergency shutdown button is hit	
	Reset equipment	
	Return to Main Menu	

My first meager attempt at writing the code structure.

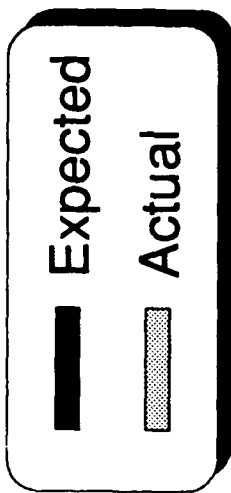
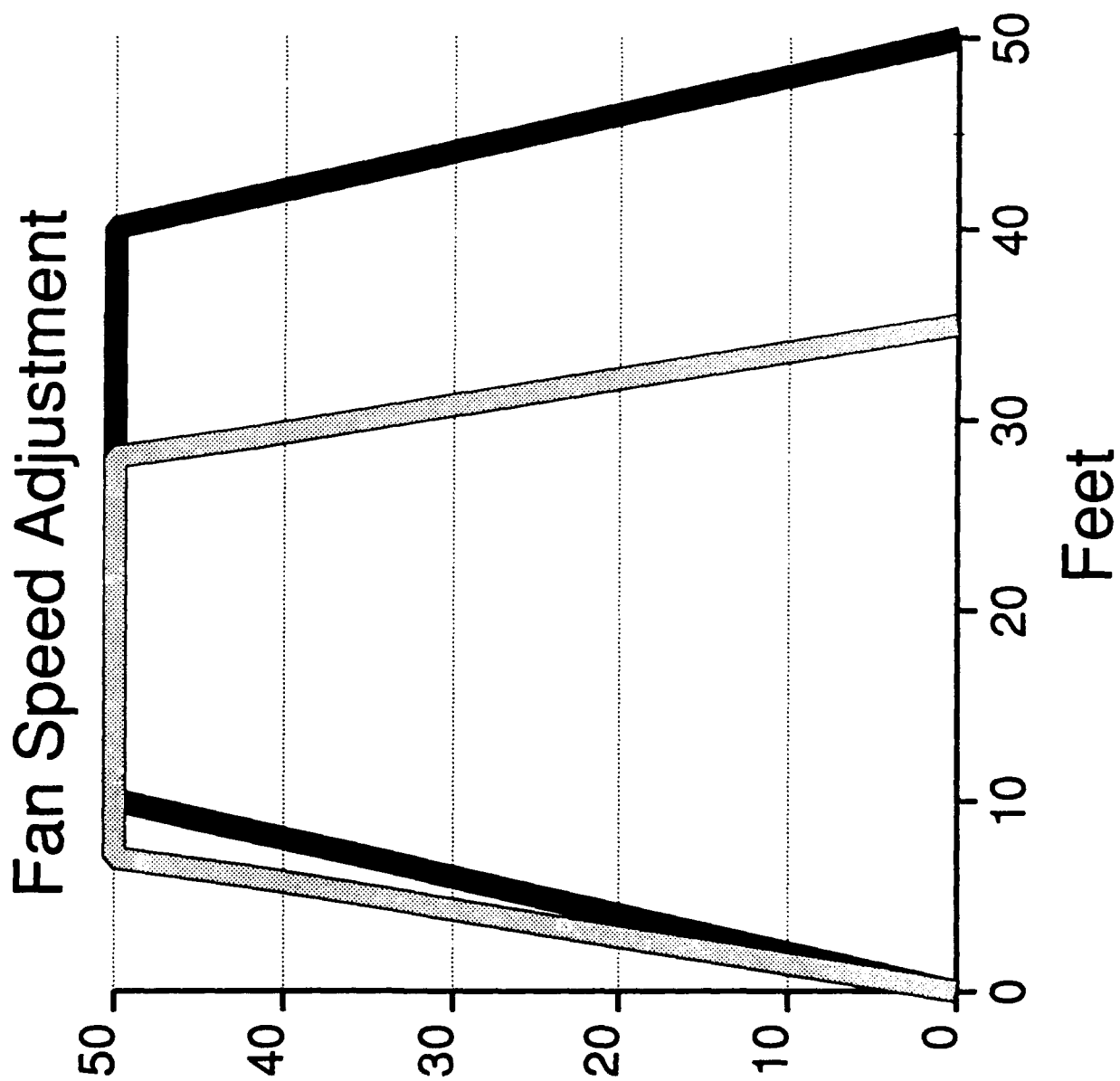
Pattern Recognition

Pump Rate Adjustment



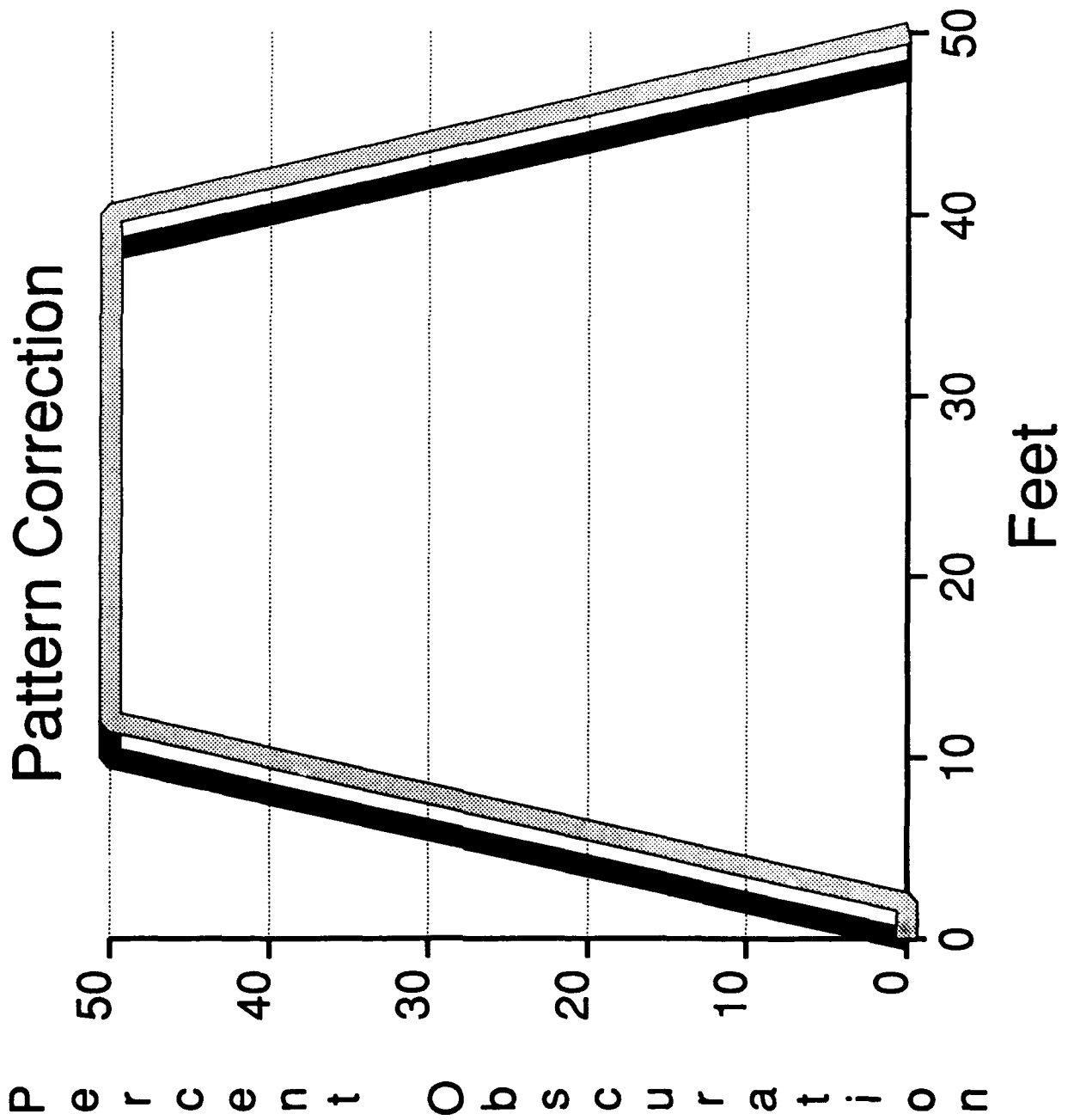
Pattern Recognition

Percent Obscuration

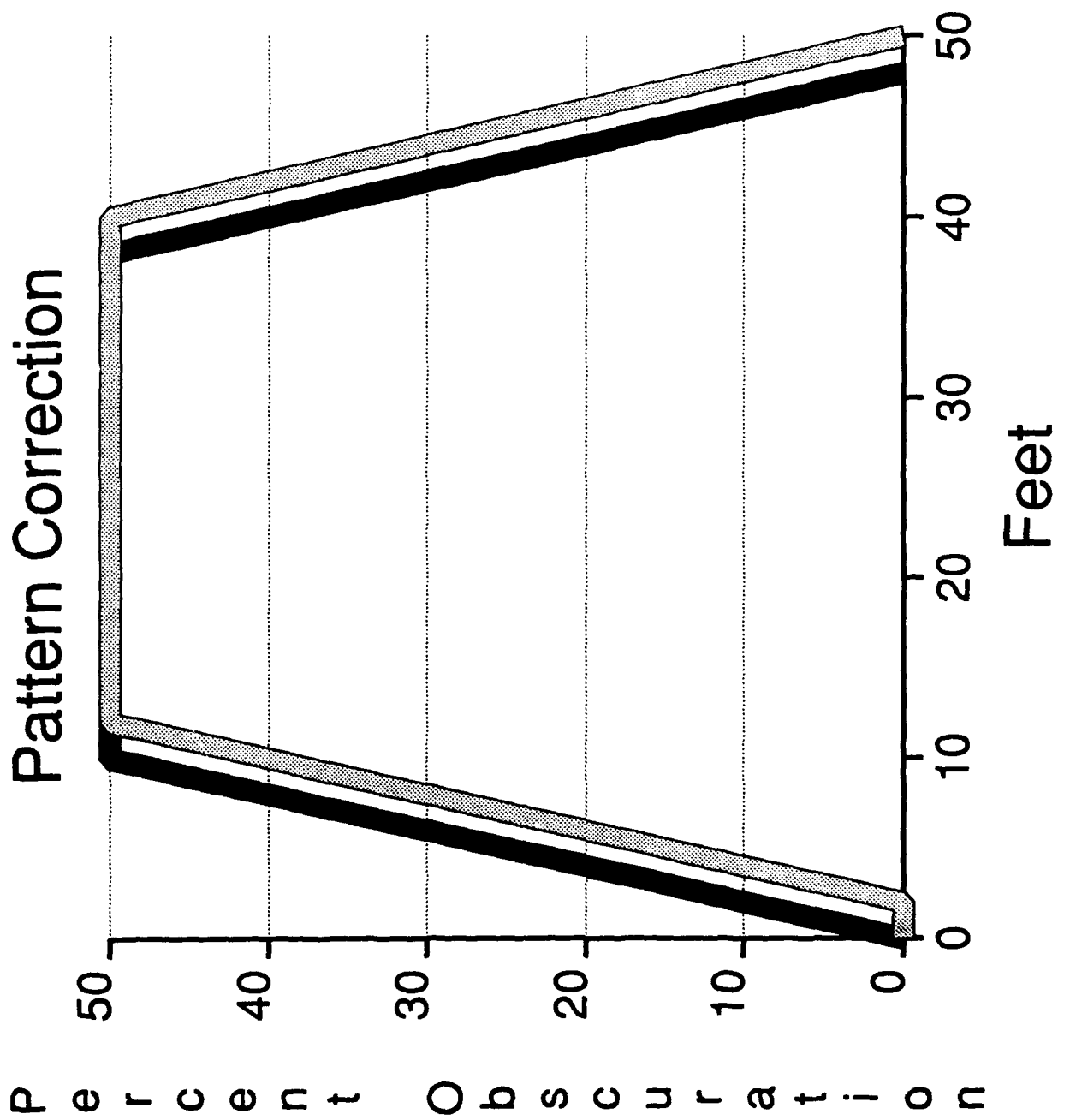


Pattern Recognition

Pattern Correction



Pattern Recognition



APPENDIX H
SOME SELECTED SOURCE CODE

"Getdata.h"

```

get_data()
{
    int temp;
    unsigned int x,y,correct=0;
    do {
        clear_screen();
        printf("Enter the number of segments in the pattern of the
aerosol: ");
        scanf("%d",&segments);
        if ((segments>0) && (segments<MAX_LENGTH)) correct=1;
    } while (correct==0);
    density [0]=0;
    for (x=1; x<=segments; x++) {
        printf("\nEnter the length for segment number %d: ",x);
        scanf("%d",&length[x]);
        if ( (length[x]<1) || (length[x]+total)>MAX_LENGTH-(seg-
ments-x) ) {
            x--;
            continue;
        }
        printf("\nEnter the density at the end of segment %d:
",x);
        scanf("%d",&density[x]);
        if ( (density[x]<0) || (density[x]>MAX_DENSITY) ) {
            x--;
            continue;
        }
        if (density[x-1]>density[x]) temp=-1;
        else temp=1;

        if (((density[x]-density[x-1])*temp)/length[x]) >
MAX_DENSITY_CHANGE) {
            x--;
            printf("\nChange in density is too high.\n");
            continue;
        }
        total+=length[x];
    }
    temp=0;
    expected[0]=0.0;
    for (x=1; x<=segments; x++) for (y=1; y<=length[x]; y++) {
        expected[++temp]=(float)((float)((float)(density[x]-
density[x-1])/length[x]) * y + density[x-
1]);
    };
    for (x=0; x<=total; x++) {
        printf("#%u -- %5f\n",x,expected[x]);
    }
    display data(&expected);
    return 1;
}

```

FINAL REPORT

Lori L. Anderson

ABSTRACT

The following report is a description of the work I performed during my participation in the Air Force Office of Scientific Research (AFOSR) High School Apprenticeship Program. At Wright-Patterson Air Force Base, I worked with a number of software packages, including word processors, spreadsheets and graphics packages. These software packages were hosted on an IBM compatible Z-248 microcomputer. As another part of my work experience, I learned how to write software in the Ada programming language. The Ada compiler I used for this effort was the Digital Equipment Corporation (DEC) Ada Compiler, hosted on a DEC VAX computer. The Ada programming comprised the majority of my work experience and the applications that I developed implemented numerous scientific and mathematical functions.

INTRODUCTION

"Make me, O Lord, thy Spin[n]ing Wheele compleate; Thy Holy Worde my Distaff make for mee [sic]." This quote from the poem "Huswifrey" by Edward Taylor compares the seventeenth-century belief of the relationship between man and his dominant God to a spinning wheel which cannot work without the hands of its master. This comparison could also apply to computers, which are electronic devices executing a series of instructions which make them perform as the programmer wishes. I have always thought of computers as helpful tools for the average human, useful but difficult to understand. Although I hadn't much experience with computers, I felt that a job supporting the Air Force at Wright

Laboratory (WL) Avionics Directorate would give me an opportunity gain insight into the world of computing. I was interested in learning how computers work and in their applications.

When I arrived at WPAFB, I found that I would be working with Marc Pitarys in the Avionics Logistics Branch (WL/AAAF), specifically with the Software Concepts Group. Because my only computer experience consisted of using the BASIC Tutorial on an Atari and using a Macintosh CAD package, I hoped that extensive computer knowledge was not necessary for this job. I was relieved to find that the main focus of my job was exactly as I had hoped, for me to learn about computers and apply my knowledge in helping the Group with its mission. I learned that it was important for me to both experience other programmer's software and write my own programs in order to gain a greater understanding of how computers work. In the past eight weeks I learned to use commercial software packages such as word processors, spreadsheet software and various CAD packages; the VAX VMS operating system; and MS DOS. However, the majority of my time was spent programming in the Ada language.

DISCUSSION

Using commercial software enabled me to become familiar with the simple workings of the computer and its operating system. The various types of software also allowed me to get to know many of the computer's input and output devices. I kept a notebook to record commands that were important to remember for each software package and to distinguish which program the commands were used in.

Because I had previously used a Macintosh CAD package in drafting class at school, I was interested in becoming familiar with other packages. I spent the first week learning to use several packages such as 3-D Drawing and CAD 3-D. Before I could use these packages I had to learn to copy disks and install them into the hard drive. One of the

first assignments that I worked on after installing the packages was to update a map of the section of the building in which we worked. An engineer in the group had created a map of the Branch using a program called CADKey and I was to revise it to correspond to the recent changes that had been made. I encountered several difficulties in using the software, but was able to overcome them by using the trial and error method. After completing the revised map, I was able to determine that the CADKey package was generally the easiest to use and produced high quality results as well.

My commercial software experience continued by learning to use both Wordstar 5.5 and WordPerfect 5.1 word processors for the PC. Once I had familiarized myself with the word processors, I was able to distinguish the strengths and weaknesses in each and determined that WordPerfect was the easier of the two to use. The options in WordPerfect were easier to access using the menus displayed at the top of the screen by the mouse and the keyboard commands were simple to find and remember.

Also, I briefly used spreadsheet and graphics software. Using the Quattro Pro and Lotus 1-2-3 software, I was able to create spreadsheets that would calculate and chart the billings, payments and earnings of various contractors. I used Harvard Graphics to create a briefing.

While using the VAX, I was introduced to computer security and also to the Language Sensitive Editor (LSE). Because I had only used personal computers (PCs), I was unfamiliar with using passwords as is common on time sharing computers, and this experience alerted me to the importance of computer security. My experience with the VAX also allowed me to use the LSE, which I used to create documents and write programs in the Ada language.

The majority of my time was spent learning to program in the Ada language. I read a book titled Ada Language and Methodology by Watt, Wichmann and Findlay which explained the history and basics of the

language and tested my skills with the programs in the problem sections. After writing simple programs from the book, I was introduced to functions and packages and thus was able to write more complicated routines.

One of the first programs that I wrote was from a problem in the Ada Language and Methodology book. It was an implementation of the game called Nim, played between the computer and a human opponent. The human chooses a number of match sticks to begin the game, then he or she and the computer take turns, each removing from one to three sticks per turn. The object is to remove the last match stick on your turn and leave your opponent with none. This program required creating an algorithm so that the computer would know how many sticks to remove on its turn. The program was a simple example of Artificial Intelligence (AI). According to Webster's New Riverside University Dictionary, AI is the display of characteristics of a machine programmed to imitate human intelligence functions. By using heuristics and the imitation of human behavior, the computer moves in response to the player's moves during the game of Nim. The algorithm allowed the computer to consistently beat its opponent unless the player made an intelligent decision on the number of match sticks to begin with and made a correct first move.

The first function that I found necessary to write was one used to convert lower case letters to upper case and vice versa. In Ada, characters are stored in American Standard Code for Information Interchange (ASCII) format. Upper case letters reside in the positions 65-90 respectively. Lower case letters have the values 97-122. Converting from case to case involves either adding or subtracting 32. The Ada statement to convert from lower to upper case is `result := character'val (character'pos (ch) - 32)`. Converting from upper to lower case used the Ada statement `result := character'val (character'pos (ch) + 32)`. These functions were extremely useful in standardizing input data in other programs.

One of the most interesting functions that I wrote was a random number generator using the linear congruence method. This method requires four fixed values to be entered. The seed (S) is the initial value used to begin the program. The equation used in my program is $R = (S * M + I) \text{ mod } D$. The seed is taken times the multiplier (M), then the increment (I), which is usually fixed as 1, is added to the total. This number is then divided by the modulus (D), which is usually fixed as a rather large number. The new value can then be re-entered into the equation, thus producing another new value. For example, in my random number function, I have chosen 729 as the modulus, 1 as the increment and 40 as the multiplier. If 25 were entered as a parameter, the seed value would become 25. The value of the equation would be $((25 * 40) + 1) \text{ mod } 729 = 271$. The first random number generated would be 271 and if the function were called again, 271 would then be entered as the seed, thus producing a new random number. Of course, there can be no such thing as a perfect random number chosen by a computer. A computer is deterministic since it executes a program that implements a set equation to determine the "random" number. The linear congruence method succeeds in returning each number an equal percentage of the time, thus in practice fulfilling the requirements for a random number. The success of the generator depends upon the numbers chosen for the multiplier, modulus and increment, therefore I also wrote a program to test any combination of these variables and determine how well the generator will work.

Because the Ada language has no predefined square root function, it was necessary for me to create my own function that could accurately calculate the square root of any positive real number. I used Newton's method to do this. The following is a description of this method. Given the equation of the parabola $y = x^2 - a$, the derivative (and slope) of this equation would be $y' = 2x$. One uses any two points on the parabola (x_0, y_0) and (x, y) to substitute into the point-slope

formula, producing $y - y_0 = 2x_0 (x - x_0)$. Solving for $y = 0$ results in $-y_0 = 2x_0 (x - x_0)$. Using the original equation, $-y_0 = -x_0^2 + a$, and substituting yields $-x_0^2 + a = 2x_0 (x - x_0)$. Simplifying produces $-x_0/2 + a/2x_0 = x - x_0$. The final equation is $x = (x_0 + a/x_0)/2$. An initial value (x_0) is entered into the program and the equation $x = (x_0 + a/x_0)/2$ is repeated until the absolute value of the difference between the square root function input parameter and the square of the computed outcome is less than an acceptable error value (0.000001).

I also used Newton's method to determine a root of any polynomial equation. The following is a description of the program. The equation would be entered, and the computer would determine its derivative. An initial value of x was entered by the user. The assignment statement $x := x - (f(x)/f'(x))$ was used to bring x closer and closer to a root. The value of the function divided by the derivative was subtracted from x , and the resulting value was then entered into the equation as x . The loop ends when the absolute value of the relative error $(x - x_0/x)$ is less than or equal to a sufficiently small epsilon distance, with x being the most recent value calculated and x_0 being the previous value. I succeeded in making the program function in a limited number of cases. The program could easily be modified to compute all of the roots by storing the degree of the equation and looping the program for each of the roots.

CONCLUSION

While reviewing some of the programs I have written, I have realized how much Ada I have learned in just eight weeks. This job has interested me in computers because I have been able to be the master controlling the computer; I have been able to see past the useful office programs and delve into the inner realm of computing, to think about artificial intelligence and the actual limits of computers in becoming more and more like humans. This job has given me the opportunity to

expand my understanding on the subject of computers and has led me to a greater awareness of possible applications of future computers.

Communication, Navigation, Identification Laboratory Research

Mr. Barry J. Koestler

Abstract

This project entailed the conversion of the User Defined Operations and Interactive Testing (UDOIT) system from use with a Hewlett Packard HP 8566B Spectrum Analyzer to use with a Hewlett Packard HP 8563A Spectrum Analyzer. Included in converting the system to use the new analyzer was reprogramming the FORTRAN codes and programs to enable them to work with the new machine. Also, the Device Descriptor Files (DDFs) were translated into useable commands for the new equipment, and new menus were set up according to the new routing of data. This portion of the project was necessary as the new machine needed different commands for its different functions and operating procedures. Also, the new machine had some functions that were not implemented in the other analyzer and thus needed to have those new functions linked to the system. Also included in this project was work in the Analysis and Evaluation Group office. This consisted of familiarization with and work on Zenith, Unisys, and Macintosh computer systems. The work included the transfer of circuit designs from their rough draft on graph paper to a finalized printout that made them easier to use, understand, and update; the creation of spreadsheets to corrolate data pertinent to office operations; and also the wiping and reformatting of disks that were no longer needed in their present form.

Introduction

UDOIT runs on the MicroVAX computers in the Communication Systems Evaluation Laboratory (CSEL) and provides a menu-driven interface for controlling hardware and software devices. It is a system designed to help automate the construction and execution of test scenarios in a laboratory environment. UDOIT allows CSEL engineers and technicians to specify, implement, and modify test conditions for a variety of units under test using automated procedures rather than manual controls. Established initially to support research and development of satellite communications, the CSEL laboratory has been modified to provide the digital and RF simulation and test capabilities required to support in-house research in the area of advanced low probability of intercept, jam resistant communications. The specialized mix of CSEL hardware and software creates a highly flexible, cost-efficient setting to explore and develop promising communication technology. CSEL is used to evaluate signal structures, signal processing approaches, anti-jam and low probability of intercept techniques and the related equipment for the Integrated Communication Navigation Identification Avionics (ICNIA) system, as well as to support development of a CNI Rapid Turnaround Support System for the Air Force Logistics Command. Through the use of the CSEL lab and the Integrated Electromagnetic System Simulator (IESS), a dynamic RF hot bench to provide realistic operational environments for development and laboratory evaluation of integrated CNI systems, the System Avionics Division conducts extensive research in the development and implementation of anti-jam Communication, Navigation and Identification (CNI) systems for USAF aircraft. The IESS facility simulates on-board avionics interfaces, real-time simulation of complex mission scenarios, and automated data collection and analysis, and when coupled with

the CSEL threat simulator, ICNIA is exercised and analyzed under realistic, controlled, dynamic signal environments to evaluate performance and identify the possibilities for implementation. The office work done included the transfer of circuit diagrams, including the "Banshee Drive," from paper to computer. This "Banshee Drive" transmits spread spectrum burst transmissions, which are then received by a spread spectrum receiver with the purpose of extracting data. The drive is composed of sixty-seven integrated circuits with a ten megacycle driver producing the spectrum burst transmissions. The work of transferring these design sketches was done on an Apple Macintosh. Also included in the work done in the office was the design and creation of various spreadsheets and data forms correlating office data to be distributed among the workers. The majority of this work was done to indicate the status of the undergoing projects and was carried out on a Unisys computer system, with some of it being done on the Macintosh. Another part of the office work was wiping disks and reformatting them for other uses. This was done both on Zenith and Unisys computers, and also through the use of a device known as a degausser, which through the use of a powerful magnet, erases the disks by organizing the magnetic memory part of the inner disk.

Discussion of the Problem

In the CSEL laboratory the UDOIT system was linked to several pieces of hardware which in turn also ran to the IESS facility. One of these pieces of hardware was the Hewlett Packard HP 8566B Spectrum Analyzer. The HP 8566B is a high-performance spectrum analyzer which operates from 100 Hz to 2.5 GHz in the low frequency band and 2 - 22 GHz in the preselected microwave band. It uses a synthesized LO to provide accurate frequency tuning and an

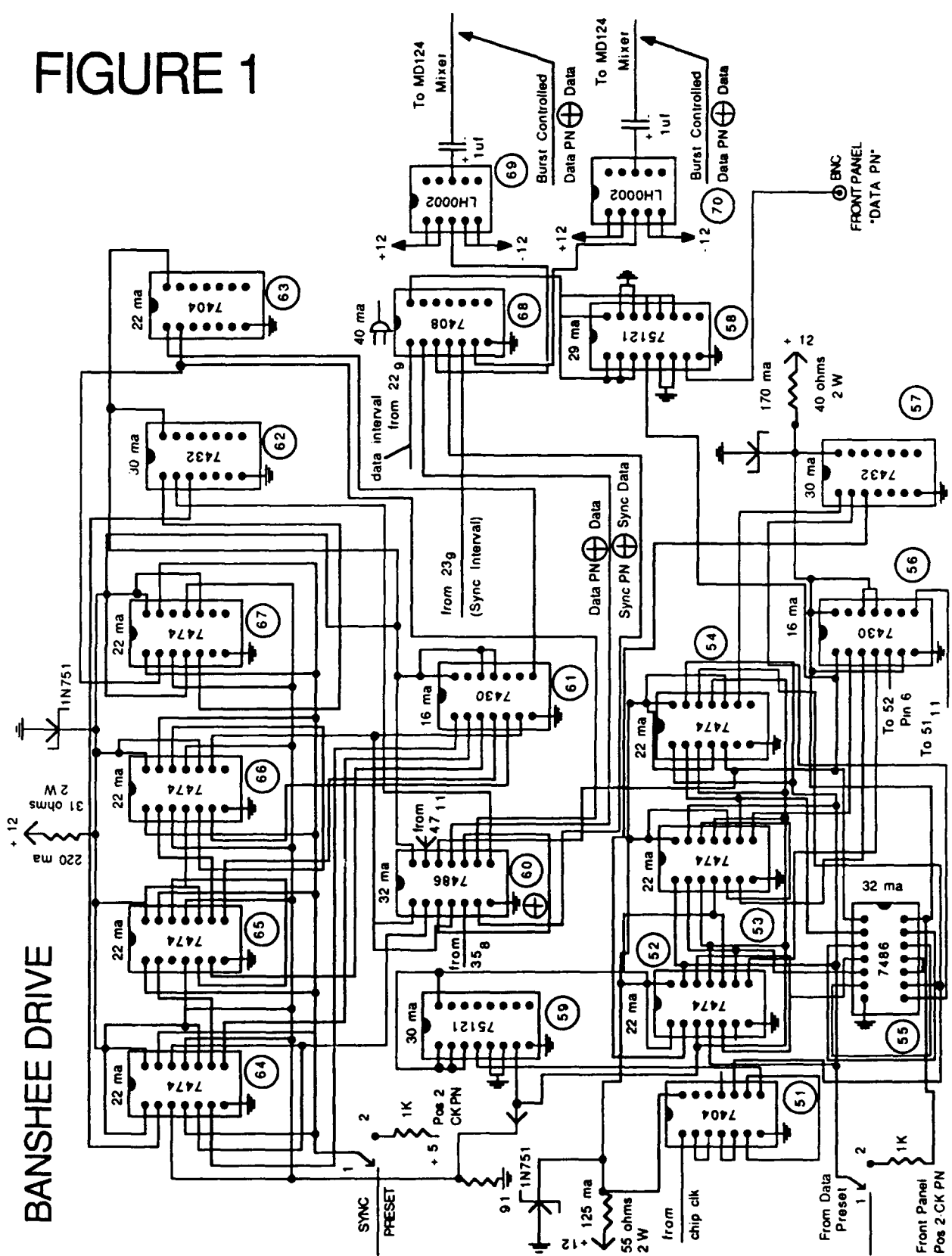
internal micro-computer to automate controls and provide useful operating features. This particular model was the one in use in the CSEL laboratory. But, when it was needed in another area for preflight tests, a new model spectrum analyzer, the HP 8563A, was implemented. The HP 8563A was a portable spectrum analyzer similar to the HP 8566B. As it was a different model, though, the new spectrum analyzer had different functions, FORTRAN codes, and data routes. Thus, it was necessary to reprogram the UDOIT system to be compatible with the new spectrum analyzer. This in turn required the making of new menus and new paths for data and commands to travel. Some of the functions of the old machine had to be erased as the new machine didn't have those functions, and also some functions of the new machine had to be added because the old machine lacked them. Because of these changes, the system needed to be reconfigured, and then completely checked to see that each menu choice worked for the new machine, and that each choice executed the proper command and followed the correct data path. Once the functions were sorted to see which were to be kept, which needed to be added, and which needed to be erased, the task of changing the menus and the FORTRAN codes began. This work had to be done with care, as the paths for the menus had to coincide exactly with that of the data, so the user could control the analyzer through the user friendly menus and not have an error. Also, if an error in the programming were to be made, it would be difficult to sift back through the large amounts of data and commands to find the mistake. After each new function had been implemented and each old function erased or put in non-implemented status, the project became one of checking each menu choice to see if it was correctly linked to both the rest of the UDOIT commands, as well as to the spectrum analyzer. The work in the office was also challenging, as I had to convert

intricate circuit drawings that were on 11 1/2" by 17" graph paper to a Macintosh program, MacDraft, and fit them on 8 1/2" by 11" pages. As no one in the office was a "Mac expert," I had to start from scratch and learn everything first hand. Soon, I was giving help to others in the office, as I had used the machine more than most of the workers. The case was similiar for the work done on the spreadsheets, as the machines that I was working on, the Unisys systems, were less than a week old, and I was the first to work with the new spreadsheet programs, Excel 3.0 and Quattro Pro. Thus, I had to learn the commands and features without help or previous use. The other part of my office work , the disk erasing and reformatting, was fairly simple, as the program that enabled the erasing and formatting of other disks was user-friendly and others in the office had used it and gave helpful instructions.

Results

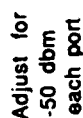
The results of this summer's apprenticeship program can best be seen through printouts of the work, as they illustrate the work completed. Figures 1 and 2 show part of the work of the "Banshee Drive" diagram that was converted from paper to computer. After these drawings were completed, they were distributed among those workers who worked with the drive and needed to know its design. Figures 3-5 show other work done in the office, including a transmitter block diagram drawing, a modem design drawing, and a spreadsheet dealing with the status of different contracts and purchase orders. This spreadsheet that I created is also flexible, as it can be used in skeletal form in which new information can be entered using the same format. The task of wiping disks and reformatting them was completed and the disks were then used for other purposes, as the old disk uses were no longer needed.

FIGURE 1



Front Panel
"Noise In"

LH0032CG



TRANSMITTER BLOCK DIAGRAM

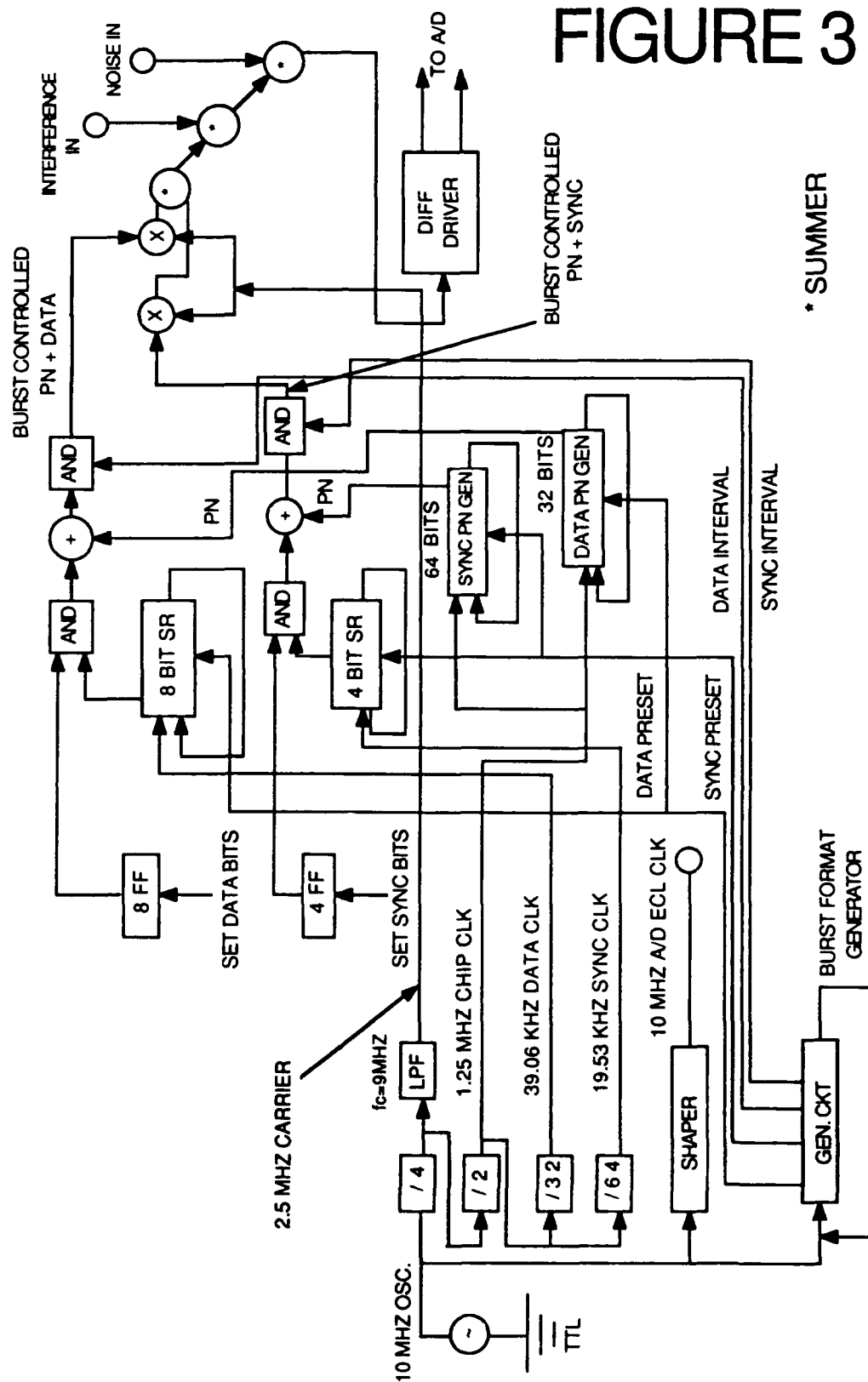


FIGURE 3

*** SUMMER**

PROTOTYPE MODEM TEST ENVIRONMENT

FIGURE 4

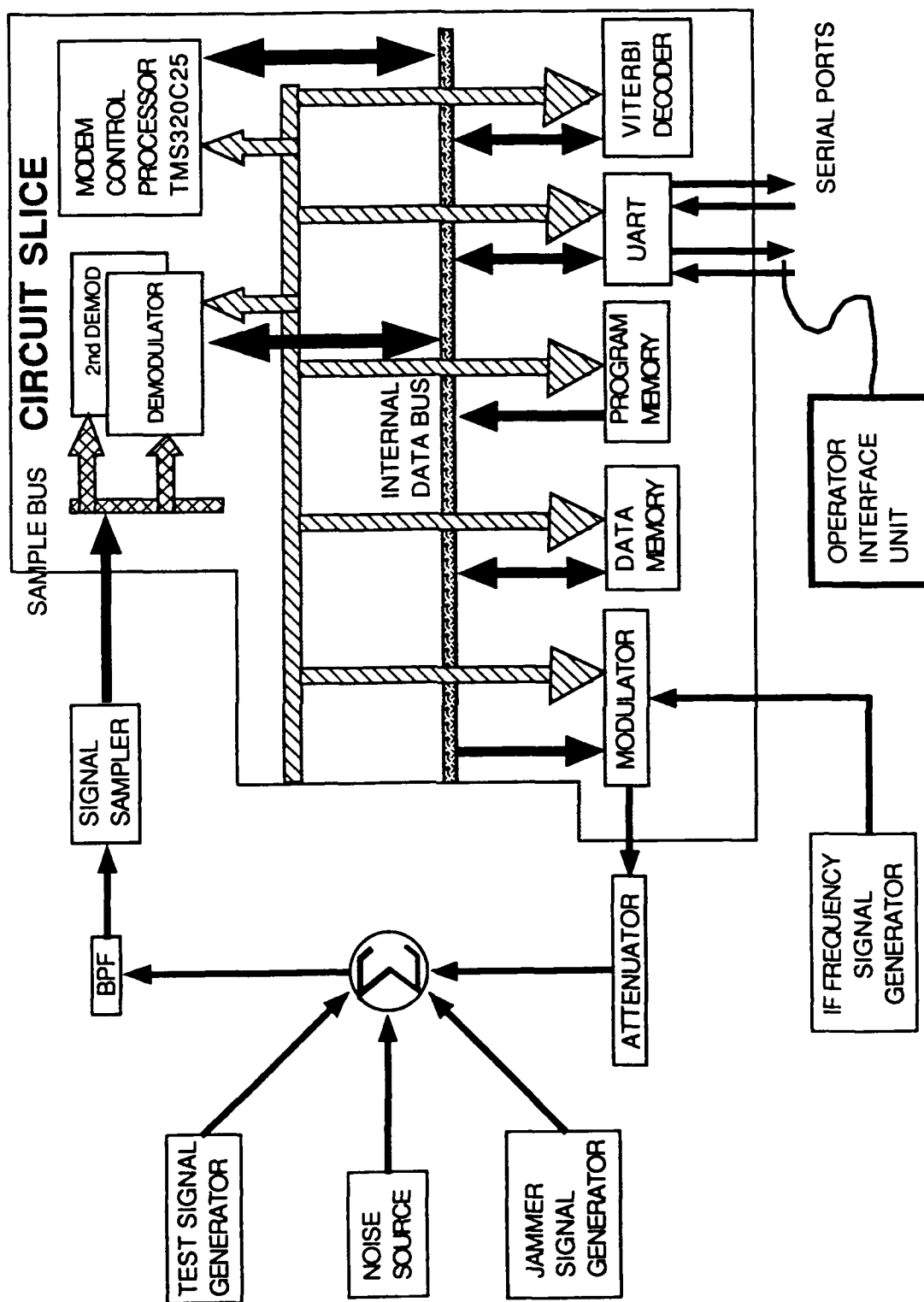


Figure 5				
Status Sheet				
AFSC Form 2399				
Engineer	No.	Title	Date Assigned	Status
Clark	7	Digital Signal Proc. Support	26-Jun-90	In Progress
	9	CSEL Support (Filter)	1-Jul-91	On Order
Connolly	2	Program Consultation	14-May-91	Awt ARC App.
Crissey	5	IESS Support	13-May-91	On Order
	6	PC Software	11-Jun-91	On Order
	7	Software Support	1-Jul-91	On Order
	8	Software Support	2-Jul-91	Hold
	9	Software Support	2-Jul-91	Hold
Falkowski	10	Network Support	2-Jul-91	Hold
	11	Software Support	3-Jul-91	Hold
	7	IESS Support	12-Jun-91	On Order
	8	IESS Support	18-Jun-91	On Order
	9	IESS Support	8-Jul-91	Closed
Harris	10	IESS Support	11-Jul-91	Closed
	38	Technical Consultation	12-Jun-89	On going
	39	Technical Consultation for Facility Upgrade	13-Jun-89	On going
	64	Install Wordperfect	4-Oct-90	Awt Software
	74	Procure Laptop PC	22-Apr-91	Due 31-Jul-91
	78	Branch R-Mail Supp.	8-Apr-91	Awt Feedback
	82	IESS Facility Support	8-May-91	Awt Feedback
	84	IESS Support	10-May-91	Closed
	85	IESS Support	12-Jun-91	Closed
	86	Wrist Straps	12-Jun-91	Closed
	87	Software Support	27-Jun-91	On Order
	88	APTEC Maintenance	1-Jul-91	On Order
	89	Purchase Oil	3-Jul-91	On Hold
	90	Order Tape Cabinet	10-Jul-91	Due 19-Jul-91
Hartman	16	Computer Hardware/Software Installation	24-Jun-91	On Hold
Howell	7	Lab Interface	29-Nov-90	On going
	8	ITB/GPSS Interface	7-Mar-91	On going
	9	Procure Engineering Workstation	29-Apr-91	Due 15-Jul-91
Rogers	8	Procure Cover Plate	11-Jun-91	Closed
Roseveare	1	Procure Workstation Furn.	22-Apr-91	Closed
	2	Procure Workstation Furn.	1-Jul-91	On Hold
Welbaum	76	Purchase Misc. Parts	6-Jun-90	Closed
Wilkins	2	Programming Consult.	19-Apr-91	On going
	3	Statistical Analysis & Support	19-Apr-91	Closed

Results(cont.)

The work done on the UDOIT system can be seen in the pages that follow. The pages are actual prints of what appears on the screen in the UDOIT system. The first step is to enter the computer system in the lab by typing the username and password. After this is completed the user may choose to enter UDOIT or any of the other functions on the system. When UDOIT is selected, the main UDOIT menu appears, and the system becomes entirely menu-driven, making it extremely user friendly. Once the machine that is wished to be used is chosen, the part of the system to be edited can be chosen, in this case, editing the Device Descriptor Files(DDFs). Then the new functions of the analyzer had to be implemented, either by adding them to already existing menus, or by creating new menus. If the functions to be added could not be placed under a similar category in a menu that had already been created, they were placed in the miscellaneous function menu, and for this reason, that menu had to be completely redesigned. After all the changes had been made, and all the FORTRAN codes and data paths had been created and checked, the UDOIT system was ready to be used with the new spectrum analyzer.

FORTTRAN STOP

\$ lo

USYSTEMS logged out at 1-AUG-1991 09:57:12.81

Local -011- Session 1 disconnected from CSEL2

Local> c csel2

Local -010- Session 1 to CSEL2 established

Welcome to CSEL2 VAX/VMS V5.3

Username: KOESTLER

Password:

Welcome to the UDOIT Automated Test Environment

- D - edit a Device descriptor
- S - edit a Scenario
- C - edit a Configuration (not implemented)
- X - eXecute a scenario
- Q - Quit UDOIT

Enter selection:

Edit DDF

- H - edit DDF Header
- M - edit DDF Menus
- E - Export your work to a file
- I - Import a file to work with
- A - Abandon your work
- G - Gateway to VMS (active in subsequent lower level menus)
- P - Print DDF
- R - Return to top level Udoit menu

Enter selection:

Select an HP8563A Spectrum Analyzer

Primary device locations

- 1 - Use Spectrum Analyzer #1 [Bus 1 Addr. 18]
- 2 - Use Spectrum Analyzer #2 [Bus 2 Addr. 18]

Secondary device locations

- 3 - Use Spectrum Analyzer #2 [Bus 3 Addr. 18]
- Q - Quit this Device

Press the space bar to continue.

Spectrum Analyzer Main Menu

- | | |
|--|----------------------------------|
| F - frequency control | C - coupling control |
| I - save/recall instrument state | P - preselector control |
| A - threshold & reference level, scale | J - RF input attenuation control |
| B - bandwidth control | E - trace math |
| S - sweep and trigger control | D - display control |
| M - marker control | T - trace processing |
| O - other miscellaneous functions | W - other trace functions |
| Z - select a spectrum analyzer | G - execute instrument preset |
| Q - Quit this device | |

Press the space bar to continue.

Spectrum Analyzer Miscellaneous Menu

A - amplitude units	S - store short
B - conversion loss	T - store thru
C - demodulation	U - sweep couple
D - signal ident. to center freq.	V - sweep output
E - signal identified freq.	W - trace data format
F - mixer bias	X - video bandwth to res. bandwth ratio
G - mixer mode	Y - video trigger level
H - normalize trace data	
I - normalized reference level	
J - protect state	
K - res. bandwidth to span ratio	
L - recall open/short average	
M - recall trace	
N - recall thru	
O - reference level calibration value	
P - squelch	
Q - store open	

R - Return to Previous Menu

Press the space bar to continue.

Edit DDF Menus

Menu title text:

Spectrum Analyzer Miscellaneous Menu

Menu name:

HP8563A_OTHER

This menu selected for export: no

DFD Count: 25

Open this menu's picked DFD ...

Trace Data Format

- A - trace data (A-block)
- B - trace data (binary data)
- C - trace data (I-block)
- D - trace data (ASCII)
- E - trace data (real number output)

R - Return to Previous Menu

Press the space bar to continue.

Video Trigger Level

- | | |
|----------------|------------|
| A - VTL (dbm) | H - VTL EP |
| B - VTL (dbmv) | I - VTL DN |
| C - VTL (dbuv) | J - VTL UP |
| D - VTL (mv) | |
| E - VTL (uv) | |
| F - VTL (v) | |
| G - VTL (w) | |

R - Return to Previous Menu

Press the space bar to continue.

Squelch

A - squelch (db)	I - squelch EP
B - squelch (dbm)	J - squelch DN
C - squelch (dbmv)	K - squelch UP
D - squelch (dbuv)	L - squelch OFF
E - squelch (mv)	M - squelch ON
F - squelch (uv)	
G - squelch (v)	
H - squelch (w)	

R - Return to Previous Menu

Press the space bar to continue.

Edit DDF Menus

Menu title text:

Squelch

Menu name:
HP8563A_SQUELCH

This menu selected for export: no

DFD Count: 14

Open this menu's picked DFD ...

Open DFD Value Descriptor

Value Descriptor type: 4

Prompt ? yes

Prompt text color: 0

Prompt text row: 0

Prompt text column: 0

Prompt text:
Enter squelch level in DBm:

Fetch value ? yes

Show value ? yes

Value color: 0

Value row: 0

Value column: 0

Value length: 16

Double precision value default:

-120.0000000000000000

Double precision value maximum:

30.0000000000000000

Double precision value minimum:

-220.0000000000000000

Open DFD Value Descriptor

Value Descriptor type: 5

Prompt ? no

Prompt text color: 0

Prompt text row: 0

Prompt text column: 0

Prompt text:

Fetch value ? no

Show value ? no

Value color: 0

Value row: 0

Value column: 0

Value length: 8

Character value default:
SQUELCH

Open Device Function Descriptor

Item number:	1	Protection:	0
Visible:	vis		
Screen address column:	1	Screen address row:	3
Selection value:	65	Color:	0
Menu line text:			
A - squelch (db)			
Device name code:	SA	Device function code:	DTD
Value descriptor count:	3	Open picked value descriptor ...	
Return action count:	1	Open picked return action ...	
Help descriptor count:	0	Open picked help descriptor ...	

Open DFD Value Descriptor

Value Descriptor type:	5		
Prompt ? no		Prompt text color:	0
Prompt text row:	0	Prompt text column:	0
Prompt text:			
Fetch value ? no		Show value ? no	
Value color:	0		
Value row:	0	Value column:	0
Value length:	2		
Character value default:			
DB			

Conclusion

Through this summer apprenticeship program, I learned much about several different computer systems, worked in a computer laboratory environment, and experienced engineers at work, which has helped me in deciding about my career plans and college goals. Also, I have been exposed to a military setting as well as an office environment and a research environment. Thus, I feel this program will help me know what to expect and what I want to do in choosing a job and a career. I have tremendously enjoyed this summer and this opportunity, and I hope to participate next year.

FINAL REPORT ON THE USE OF PROLOG AND RULE BASED SYSTEMS IN ARTIFICIAL INTELLIGENCE

Research Apprentice, Allen L. Lefkovitz

My position this summer in the Artificial Intelligence Division of Wright Laboratories, at Wright-Patterson Air Force Base has presented me with many of the basics and uses of Artificial Intelligence (AI). My research this summer consisted of three areas: basic hierarchy, creating an expert system using a rule based shell program, and programing with Prolog. While the first four weeks of my job involved learning the purposes and concepts of both basic hierarchy using Procedure Consultant and rule based systems using Personal Consultant Plus, the last four weeks were devoted to programing with Arity Prolog. The first two areas of study were basically the ideas that are combined and utilized in Arity Prolog. In working with basic hierarchy, I created a consultation program involving choosing the right college. My work with a rule based shell program involved drug interactions with medicine for diabetes. In Arity Prolog, I dealt with the enhancement of a demonstration program involving a mechanical gripper arm that can pick up blocks and stack them according to the program user's commands, typed in plain English.

Artificial Intelligence, the research involved with supplementing human intellectual abilities with a programmed machine, is a rapidly growing and changing field. Artificial Intelligence, conceived in 1950 by Alan Turing, involves general and expert problem solving in a

generative and comprehensible language. Computer programmers are working on creating an AI system that can manipulate a domain of knowledge to gain the perceptions of hearing, speech, and vision.

My first project, involving basic hierarchy, was creating a simple tree diagram that a person could consult with for help in choosing the right college. Depending on the user's answers to different questions, including questions that show the user's preference of size of the student body, location (in/out of their home state), the variety of majors, and cost, a suggested type of school would be presented to the user. This program would then ask the user if they would like to enter into another consultation involving the different types of financial aid available.

Using a rule based shell program to create an expert system, my second project allowed me to program a consultation system and its knowledge base. The shell program provided both helpful and organized functions that are not present in normal if - then programming. This project also gave the opportunity of applying artificial intelligence to my planned future study of pharmacy by creating an expert system that would help a doctor in prescribing medication to a patient who is a diabetic. Similar to my hierarchical project dealing with choosing a college, this expert system would ask the user, a doctor, different questions concerning the patient's sex, health, and type of diabetes, whether a (female) patient is pregnant, any medication the patient is already taking, and the medication and dosage that the doctor is considering prescribing to control the patient's diabetes. Using The Physician's Desk Reference, I entered different rules concerning the main drugs used to treat diabetic patients (Insulin, Chlorpropamide,

Tolazamide, Tolbutamide, Glipizide, and Glyburide) and the type of patient that should take them. I also used The Physician's Desk Reference to make rules that would warn the doctor of too large or too small a dosage and of possible interactions between any medication that the patient is already taking and the medication the doctor is considering prescribing. Interactions between drugs involve a dangerous increase or decrease in the effectiveness of a drug and is an area of study of increasing importance.

After finishing the drug interaction program, I began working with a more difficult form of expert systems. This expert system required me to begin learning the fifth generation computer programming language of Prolog. Unlike my previous computer programming experiences, Prolog is more of a then - if style of programming. To begin learning the basics of Prolog, I wrote a simple program that would tell the user who and how people in my family are related. The knowledge base contained very basic information such as a person's name, sex, a parent, and spouse (if any). Then, a variety of rules are used to define various other family relations. Some of the relations that the program could discern included the following: father, mother, brother, sister, cousin, aunt, uncle, mother-in-law, father-in-law, grandparents, great- and great-great grandparents, niece, and nephew. This program of my geneology covered over four generations of my family and also included my brother-in-law's family. The final part of this program allowed the user to even see who in my family are not related to each other.

My next project to help me get more acquainted with programming in Prolog was to transfer one of the hierarchy trees from Procedure

Consultant into Prolog. This, unlike my geneology, was a consultation program that uses the input to diagnose why a car will not start.

My final and more indepth project involved major rewriting and enhancements of a program named BLOCKSIM, previously created by Major Glen E. Monaghan, with whom I worked this summer. This program originally involved four, one unit high square blocks and one, one unit high wedge (pyramid) block on a table with space for up to five piles of blocks. The user could move the blocks around with a mechanical gripper arm.

The first change that I made to the program was to make it possible for the blocks and the wedge to be higher than one unit tall. This change created a need for rules that prevent stacking a pile above five units high, so that the gripper can move within its six unit high boundary, and rules that tell the user that a pile between the source and destination piles would be too tall to safely move the source block. This enhancement also required a change in the commands or predicates that handle the graphics for the simulation.

The second enhancement to the program was to add the user command of "move" to the English command repertoire consisting of "stack" and "unstack." The "stack" command tells the gripper to place one block on top of another, while the "unstack" command takes the specified block off its current stack and places it on the table. In either case, the rules take into account whether there are blocks in the way that must be moved before the gripper can do what was requested. The new "move" command, unlike the "stack" command, allows the user to specify the destination of a block as either the table or the top of a specific pile, rather than on a specific block. Because of the increased complexity of the "move"

command, a second set of higher-order (meta) rules, similar to those for "stack", had to be added to the program.

The third enhancement made to this program, was to change the way the gripper moved to "stack" or "move" blocks. In the original program, the gripper would start at the top of the screen and move horizontally until it was above the pile of the source block, move straight down to the block, close the gripper, move the block straight up to the top of the screen, move horizontally over to the destination pile, and then lower to the point of being one unit above where the block is to be set before opening the gripper and releasing the block. This enhancement involved measuring and totaling the heights of all blocks in a pile, and comparing that total height to the other pile heights between the source and destination to determine exactly the minimum height the gripper must move up to keep from hitting any intermediate pile, rather than moving all the way to the maximum height every time.

The next enhancement involved adding additional blocks on top of some piles. This added an entirely new dimension to the program, because this meant that there were more blocks than table spaces. This also means that there will not always be an empty table space for every block when attempting to clear off either the source or the destination. This created the problem of an unending loop when most of the previously written rules were executed. A rule had to be added that would allow unstacking onto the top block of a pile if no table space was available. This new rule created new problems, such as attempting to create piles that are too tall or building tall piles in the center of the table that prevented much movement from one side to the other.

Another enhancement made to this program was the addition of two more user commands that combined the movements of "stack" or "move," depending on whether the user's destination point was a block or a pile. These commands were "put" and "place." Just as the addition of the "move" predicate, this required a new set of higher order rules. Because the various English commands appeared to be similar, but required different objects ("stack" only stacks onto blocks, while "move" can stack onto piles, and "put" can do both), we needed to help the user understand what commands were available and how to use them properly. The next enhancement to this program was the addition of more and clearer assisting statements that recognize invalid commands and tells what commands are known, or how to use them, whichever is appropriate.

As more and more enhancements were made, many new problems arose and rules needed modifications. One of these problems was that in certain arrangements of blocks, the gripper would actually not raise the block high enough, thereby hitting another block, or it would try to pull a block out from under another. To solve this, the rules had to be modified so that any blocks above the source and destination blocks should be removed first.

The largest problems encountered involved the case where both the source and the destination blocks were in the same pile but either there were blocks separating them or the destination block was above the source block. In these cases, rules were created that first placed the source and destination block in different piles and removed any interfering blocks to some pile other than those of the source or destination blocks.

This enhancement prevented unrelated blocks from being moved to a place where they would interfere with the user's command.

The projects which I have done this summer seem to have brought more order into my thoughts during problem solving, while also teaching me many techniques in programming. A major factor in debugging my program, finding and fixing any problems, was my style. One thing in Prolog that was a necessity to learn was how important correct spacing and punctuation are to the order and proper execution of a predicate. Originally, I typed my statements all the way across the screen before going to the next line, but I was shown that starting a new line after most punctuation marks and indenting the thoughts as they change in level of difficulty, it was easier to find errors and to discover the function of a predicate. I also learned that in longer predicates, keeping the variable names consistent is vital to execution and debugging. The use of recursion and application of previously-written predicates was very important, especially in my enhancements of the mechanical gripper arm movement, in finding the top of a pile, and in the stack and metastack predicates. Another important factor in my programming was the placement of initial information that determines whether the right predicate is being used. By checking for essential and unique characteristics first, it is much easier to determine which predicate is to be used to carry out the user's commands.

Even though they are essentially complete, the tedious yet insightful enhancement of this program and my work with basic hierarchy and expert systems helped me gain both invaluable experience and knowledge of artificial intelligence, and a glimpse of the future.

(PROGRAM LISTINGS ON FILE AT RDL.)

SUMMARY OF RESEARCH ON A NEW RADAR SCENARIO

Meredith A. Lewis

The research that was done in the term of eight weeks consisted of a brief report on the history of airborne radar and an in depth research and development of a new radar scenario and simulation. The task was to design and simulate a passive radar scenario from very little prior knowledge of radar.

The research program began with a one week period of time to produce a report on the history of radar (a sampling of this 15 page report can be seen on the next page). This task was beneficial because it required the use of the MacIntosh computer, which was useful in the larger project. In the following six and a half weeks, work time was devoted to the project of designing a passive doppler radar design. The task was to evolve various geometries for a variety of scenarios and to produce numerous original equations from basic knowledge of the doppler shift.

The project that was undertaken this summer, was to work out the configurations and equations to go along with a passive doppler radar system. Then with this information, a simulation was developed an executed, validating the work that was done for the geometry and



1904. Christian Huelsmeyer was granted a German patent for an anticollision device using continuous wave (CW) radar to detect ships and trains in fog (4,5).



1940. The Bristol Beaufighter with radar detecting targets at a 3 to 4 mile range, was the first successful radar equipped fighter. It made its first kill on the night of November 7, 1940 (5,5).



1964. This unusual aircraft was the first interceptor design to be equipped with a pulse doppler radar, the Hughes ASG-18. The usefulness of this radar in the YF-12 Blackbird caused it to be refined and incorporated into the current line of modern Hughes radars for the F-14, F-15, and F-18 fighters. (5,268).



1991. Two E-8 Grumman Joint-STARS aircraft were sent over to Saudi Arabia from a test program for use in the war. While behind friendly lines its sensors can detect, locate, classify, and track moving and fixed targets (1,58). January

equations. I served as the principle researcher and mathematician for the project, while another high school apprentice served as the computer programmer for the simulation. The geometry for this radar was a bistatic scenario. The scenario involved a ground transmitter, a ground target, and an airborne receiver. First of all, a basic understanding of doppler was needed in order to derive the proper equations for the simulation. The topic of doppler shift and the doppler equation were researched in depth, and from there this concept was expanded upon independently until the proper geometries, derivations, and equations were discovered. After one set of equations was discovered for a particular configuration, the equations were modified to allow for variations in the configuration of the components of the radar. Then when diverse geometries could be applied to one complete set of equations for the doppler shift of the received signals, research was done on a small target study. This study involved researching the velocities of moving ground targets, such as cars and trucks, and computing their doppler shifts for four different radar bands (H, C, L, and Ku) using some given data. This study was done to investigate the possibility of filtering out such useless

frequencies from the receiver. From this point, further research was done on the geometries and the equations that were derived from these configurations. After researching many pieces of technical literature and much independent thought, two alternate forms of the equation were discovered that were perhaps more accurate. These alternatives were compared to the original equations using PC - MATLAB. This programming language was also learned in the course of the summer.

After the alternatives were discovered to the original equation and they proved to produce very similar results, several graphs of the working simulation were produced, also using MATLAB. The graphs were made for the doppler frequency of the target and the amplitude of this frequency. In addition to the original scenario, another set of equations, geometries, and MATLAB programs had to be worked out for a different kind of target motion than what was in the original scenario. This also proved to be a challenge. When all the programs were running smoothly in MATLAB and no further research could be done on the equations, the whole scenario and equations were given to the computer programmer, who in turn programmed the simulation with graphics, data

displays and a few other windows.

The results of this study were a technical report that has my mentor's and my name on it, which describes thoroughly the equations, derivations, and geometries from the very basic doppler equations that are found in literature. This report was turned in to begin the publishing process on the last day of the apprenticeship program. This report is well over 50 pages. This report only includes the research and mathematics that were involving in producing the simulation for the radar scenario. Through much research, independent discoveries that are not documented in other literature, and detailed mathematical derivations a systems of geometry and equations were developed that can be put to use in a real world radar scenario.

This task was very challenging as well as rewarding for me. The hard work and effort that was put into the process of developing this radar scenario really paid off as it all came together within the last week of the program. It is very possible for this radar to be developed even further, and to actually be put to use. The possibilities of expansion and use of the radar scenario are endless, and its applications are very useful for

covertness. Hopefully, one day this can be utilized in a real world situation.

RADAR STUDY

Eric Powers

Radar has many applications that have not been touched upon. I wrote a C++ program that demonstrated a new kind of radar. The group in which I worked was composed of myself, Meredith Lewis, and Greg Power. All of our efforts contributed to the end demo program.

The concept of radar was introduced to me in my first week by means of a report. I searched through the WPAFB Technical Library and found pictures of major accomplishments in radar history. I then scanned these images into a Macintosh computer and captioned these images. Thirty pictures made up the content of the report. An example is provided below.



1886. Henrich Hertz demonstrated that radio waves reflect off metallic bodies.

The problem my group was faced with concerned a certain form of radar. We were to produce a demo program displaying how the radar would actually work. Meredith, an associate, researched and derived the equations which my

program uses. My task was to learn the Borland C++ programming language, and write this radar demonstration program.

Having ~~no~~ prior C programming experience the major task was learning the language. I spent the next week reading the C++ manual and trying there examples and writing little programs of my own. Greg gave me my first task. I was to write a radar display program for someone in the office to use. After completing the first task, Greg told me to write the radar demo program. This program on completion had 12000 lines of code.

After the completion of the demo program code, my group gave a briefing on the development of it. We briefed the branch chief and several other civilians. Greg and Meredith described how the programs equations were derived and I explained and gave a demo of the program.

**COMMUNICATION, NAVIGATION, IDENTIFICATION LABORATORY
RESEARCH**

BY

MS. KAREN THOMAS

**RESEARCH AND DEVELOPMENT LABORATORIES
HIGH SCHOOL APPRENTICE PROGRAM
SUMMER RESEARCH PROGRAM
AUGUST 6, 1991**

COMMUNICATION, NAVIGATION, IDENTIFICATION LABORATORY RESEARCH

MS. KAREN THOMAS

ABSTRACT

The tasks that I encountered as a research apprentice were varied because of the nature of the organization I worked under, which was the Analysis and Evaluation Group. I had numerous opportunities to interface with engineers, scientists, managers, and electronic technicians. My assignments centered around work being done in the Communications Systems Evaluation Laboratory, (CSEL), and a smaller in-house laboratory. While in CSEL, my work revolved around the Spectrum Analyzer. In the smaller in-house laboratory my work dealt with creating designs for engineers on the Super Macintosh Computer. Along with working on the Macintosh, I used a Zenith computer system to convert documents from the WordStar Program to the Word Perfect Program.

INTRODUCTION

The Analysis and Evaluation Group basically does research and development (R&D), in support of Communications, Navigation, and Identification (CNI) projects. Being basically an in-house laboratory, the jobs and tasks of the group are spread out over a large field of topics. Working in such a group gave me an opportunity to work with different people with a variety of occupations.

While working in the CSEL laboratory, a Communications System Evaluation Laboratory, I worked alongside engineers and contractors performing operations with different machines. CSEL was originally used to

support R&D of satellite communications. Now its major function is to provide test capabilities that are needed to support tactical communication systems. It is used a great deal to support the development of a CNI Rapid Turnaround System for the Air Force Logistics Command and the Integrated Communication, Navigation, Identification Avionics (**ICNIA**) program. The laboratory is used to investigate the structure of signals, it processes the approach of signals, and it also investigates ways to create anti-jamming of signals and to create low probability of intercept techniques. The Communications System Evaluation Laboratory along with the Integrated Electromagnetic System Simulator (**IESS**) are two R&D facilities that were developed for testing and evaluation of Communication Navigation and Identification (**CNI**) systems for the United States Air Force aircraft that have to operate in the highly dynamic electronic battlefields of the sky.

Part of my assignment while working in CSEL was with the Spectrum Analyzer. The Spectrum Analyzer is a machine that uses a synthesized LO to provide accurate frequency tuning and an internal micro-computer to automate controls and provide useful operating features. The Spectrum Analyzer can be operated manually or by using the computer keyboard. The basic controls consist of FUNCTION keys and DATA control keys. By using different keys, different commands are transmitted to the machine. These different commands create different functions that will then appear on the cathode ray tube (**CRT**) graticule field. The function values are shown as they are changed by manipulating the knobs manually or as they are changed by working with the keyboard.

The CRT displays:

- active function name and value
- graticule
- values that calibrate the frequency, time, and amplitude axes

- values for the spectrum analyzer (**COUPLED FUNCTIONS**)
- operator originated labels and graphics

After a function is activated, it is immediately read out into the graticule. Its name along with the present value is immediately written in the active function area as well as the present value. ***(FIGURE #1)**

The FUNCTION keys enable a function to be displayed in the active graticule area of the CRT and on the outside graticule border. The value of the active function can be changed by using either the DATA knob, step keys, numeric keypad, or a combination of all three. If a certain state of data needs to be held in that position then the HOLD key above the DATA key can be used. This prevents any unwanted data from being added. This same key can also clear the previous function.

The DATA controls are used to change the function values for certain functions such as the center frequency, start frequency, resolution bandwidth, or the marker position. These controls (**DATA**) are around the FUNCTION keys which activate the controls that are used the most. The DATA controls will change the function in a manner in which the function prescribed it to be changed.

While in the smaller in-house laboratory I created designs on the Super Macintosh for different engineers. The Super Macintosh has a variety of tools and programs available to make any job easy to do. Some of the designs were tables and graphs, but the majority of the designs were schematic drawings of boards for radios and computers. For example, on one specific drawing I had to use many different tools to complete the job. ***(FIGURE #2-5)** Some of the different options that are available are the color keys, font sizes, format choices and other extras like circles, arrows, rectangle.

Along with working on the Macintosh, I used the Zenith to convert docu-

ments from the WordStar Program to the Word Perfect Program. This was done by going through the following steps:

- (1) Go to convert info. from Drive B to Drive A
- (2) Go to Source format -W.S. (Word Star) 3-3 from target Wir 5.1
- (3) Go to selection source B: \
- (4) Source file specify (type in)
- (5) Target A:\ (type in)
- (6) Esc. (Escape) - PF:\
- (7) Begin Conversion

DISCUSSION OF PROBLEM(1)

During my apprenticeship, I was given a number of problems to work with. Due to the nature of the organization it was necessary to be given a number of jobs instead of just one specific job. The main job I worked with was my work on the Spectrum Analyzer. Using the Spectrum Analyzer dealt with creating, editing, and running different scenarios on the UDOIT (USER DEFINED OPERATIONS AND INTERACTIVE TESTING) software system. My lab instructor gave me a list of functions to run on the machine. I was given instructions to change certain functions like the center frequency, frequency span, and stop frequency. ***(FIGURE 6)**

For example, in order to set the center frequency to 9.0 GHz I could either set it manually or by using the keyboard. By manually operating the machine I had to press a button labeled "CENTER FREQUENCY," and use the knob, step keys, or the numeric keypad to tune the center frequency to 9.0 GHz. After this is done the activated function (**CENTER FREQUENCY**) appears on the CRT graticule field. In order to set the frequency span to 1.00 Mhz I had to press a button labeled "FREQUENCY SPAN," and use the knob, step keys, or numeric keypad to set the frequency span to 1.00 Mhz. After that was completed I had to save my changes under my user name. Saving the changes insured that the scenario could be run at a later date. After the saving of the information was

complete I had to run the scenario again to make sure the changes were done correctly. If there was a fault in the editions, an error marker would appear on the screen.

RESULTS(1)

After all the input and saving of information for the center frequency was completed and tested the results appeared on the CRT graticule field.

***(FIGURE 7)**

After all the input and saving of information for the frequency span was completed and tested the results appeared on the CRT graticule field.

***(FIGURE 8)**

CONCLUSION(1)

The results came to show that all of the input information was correct and the operations ran correctly for the center frequency and frequency span as shown on ***(FIGURE 7 & 8)**.

DISCUSSION OF PROBLEM(2)

The next stage of my laboratory work with the Spectrum Analyzer was converting Device Function Descriptors (DFD) and Value Descriptors from HP8566B to HP8563A. The descriptors needed to be converted because the new Spectrum Analyzer had different commands, operating procedures, and functions. The new functions were also added into the program to make the hardware useable. The new machine had functions that were not programmed into the system for the old machine. This was done by changing characters from the old Spectrum Analyzer to the new Spectrum Analyzer. After this is done each picked value descriptor (screen from UDOIT menu) character default value should be changed to correspond with the new descriptor character. After all of those changes are made, then editing is done to make sure all of the new characters correspond to the old characters. Then, when all

of this is done the changes are saved.

RESULTS(2)

saved and tested the results appeared on the CRT graticule field. If something is wrong with the changes made then an error would appear on the CRT graticule. If an error did appear, then an Error Correction Routine could be run.

***(FIGURE 9)**

CONCLUSION(2)

The results came to show that all of the input information was correct and the operations in the new Spectrum Analyzer ran correctly for all of the new and edited functions and commands.

DISCUSSION OF PROBLEM(3)

The first stage of my in-house laboratory work dealt with working with the Super Macintosh Computer to create schematic drawings of boards for computers and other equipment for engineers. I was given engineering drawings and assigned to recreate them on the computer. Some of the designs required lots of thinking as to which tools to use. The Mac had lots of options to choose from. There were many different assortments of tools from "spraycans to paintbrushes" and from "rectangles to ovals."

RESULTS(3)

After the designs were completed and edited, completed, numerous times, they were given to the engineer. Some of the computer programs had to be manipulated into uses other than their intentioned use. This was done because some of the tools that were needed were unavailable. For example, there were no tools for coils on the program so a series of circles had to be used in their place. Also there were no tools for breakers in the program, so by manipulating the "line" tool, the breakers were made.

CONCLUSION(3)

The results came to show that the drawings were properly designed and edited as shown in ***(FIGURE 2-5)**

DISCUSSION OF PROBLEMS(4)

The next stage of my in-house laboratory work dealt with working with the Zenith computer system. I was given an assignment to convert documents from the WordStar Program to the Word Perfect Program. This was necessary because the Word Perfect Program had more options and was being used to replace an old version of the WordStar Program. Word Perfect was also more accesible than Word Star. As once stated, this was done by going through the following steps:

- (1) Go to convert info. from Drive B to Drive A
- (2) Go to Source format -W.S. (Word Star) 3-3 from target Wir 5.1
- (3) Go to selection source B:\
- (4) Source file specify (type in)
- (5) Target A:\ (type in)
- (6) Esc. (Escape) -PF:\
- (7) Begin Conversion

RESULTS(4)

After the conversions were completed I ran the file list to make sure it was blank in the old disk and made sure it was filled in the new disk. Because of the easy to follow instructions there were no complications from the conversion series.

CONCLUSION(4)

After running a few files the results came to show that all of the directions were done correctly and all of the conversions were running properly.

ACRONYMS

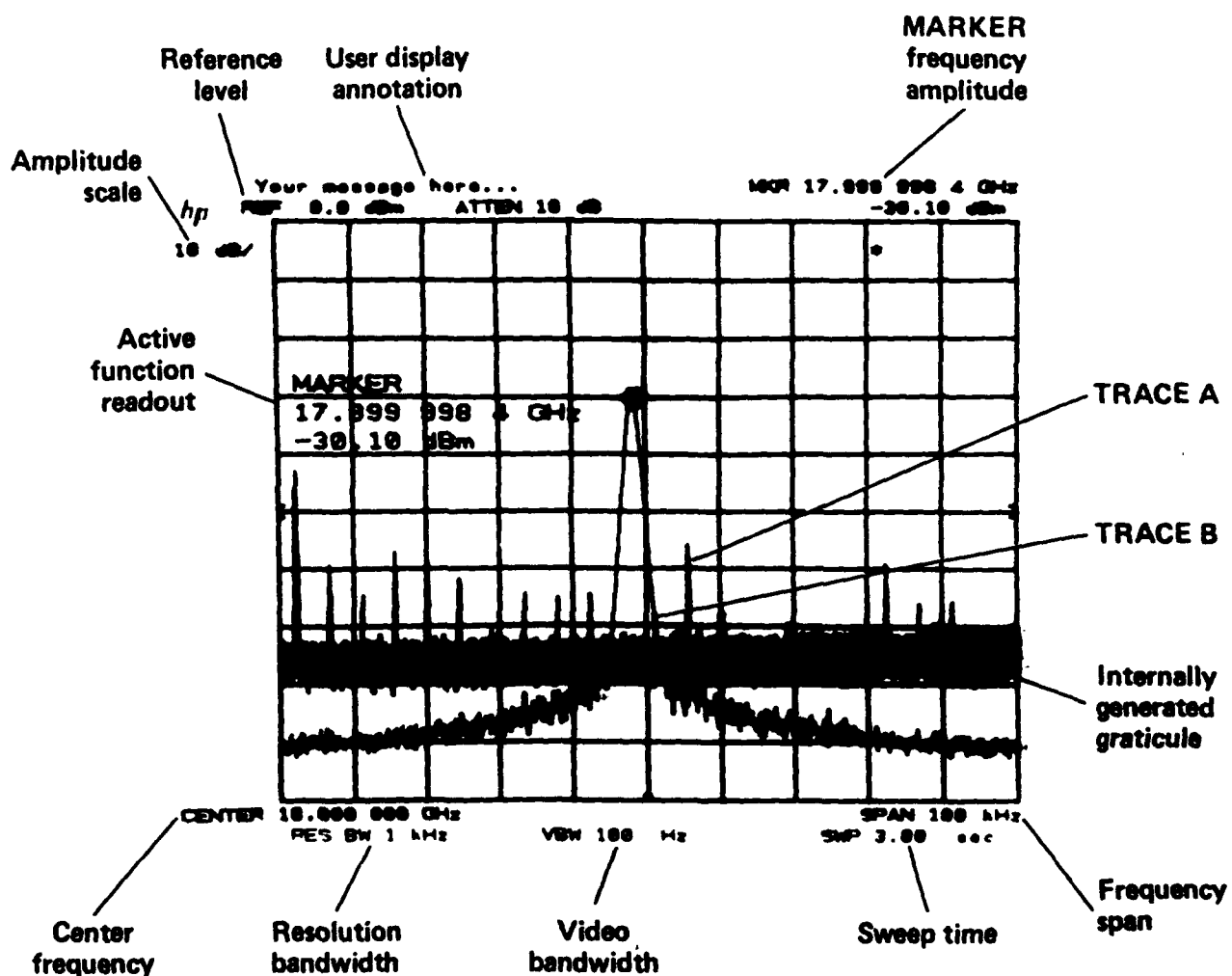
Before I could get started working with my different jobs, I had to learn the office language. Mainly the acronyms used daily around the office. There are so many different programs going on around the office, so many of them are

given acronyms to cut down on confusion. Learning the different acronyms made the task of learning the different programs and jobs easier. ***(FIGURE 10)**

* (Figure 1)

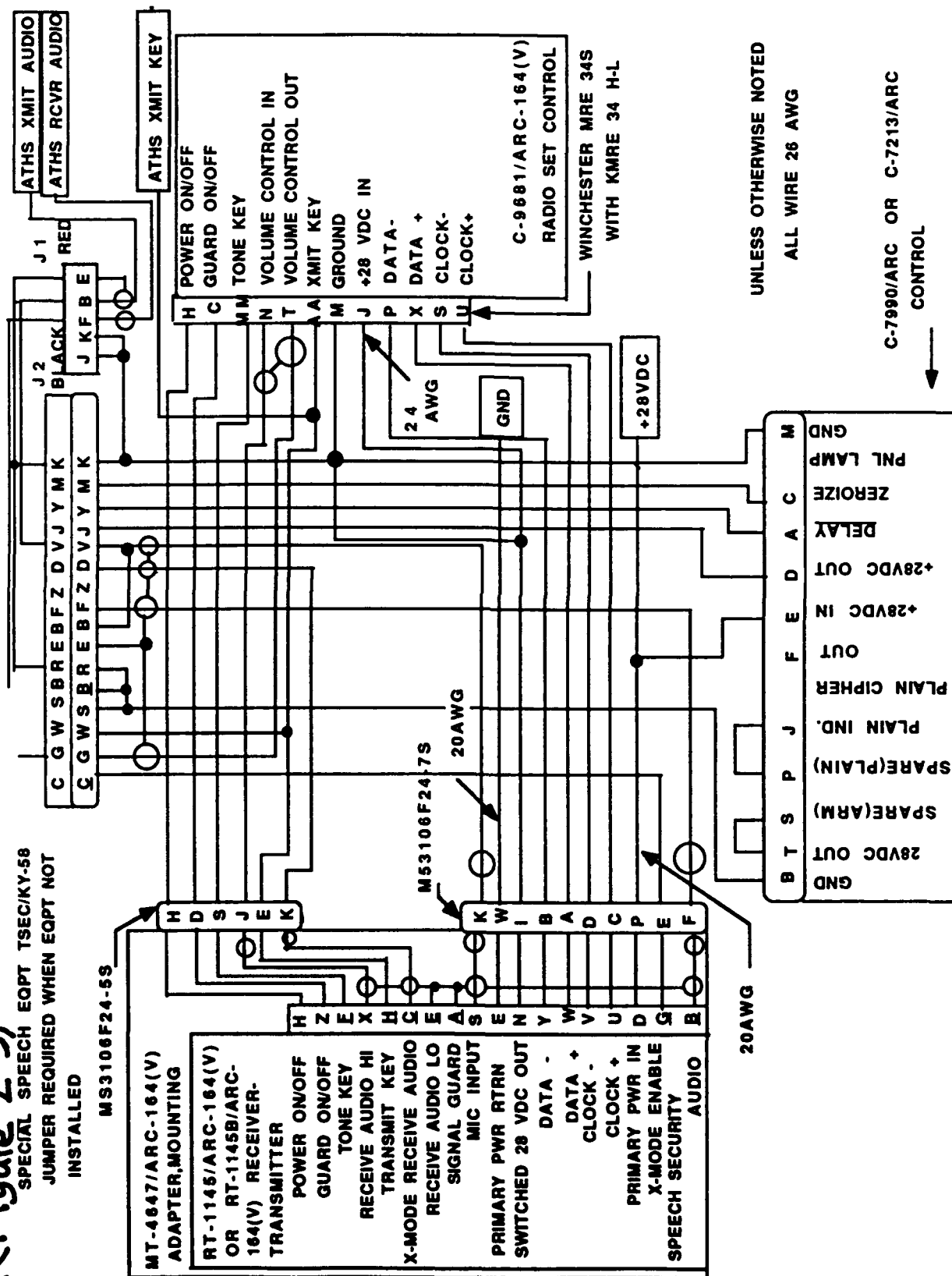
CRT DISPLAY

The analyzer's CRT display presents the signal response trace and all pertinent measurement data. The active function area names the function under DATA control and shows the function values as they are changed. All the information necessary to scale and reference the graticule is provided.

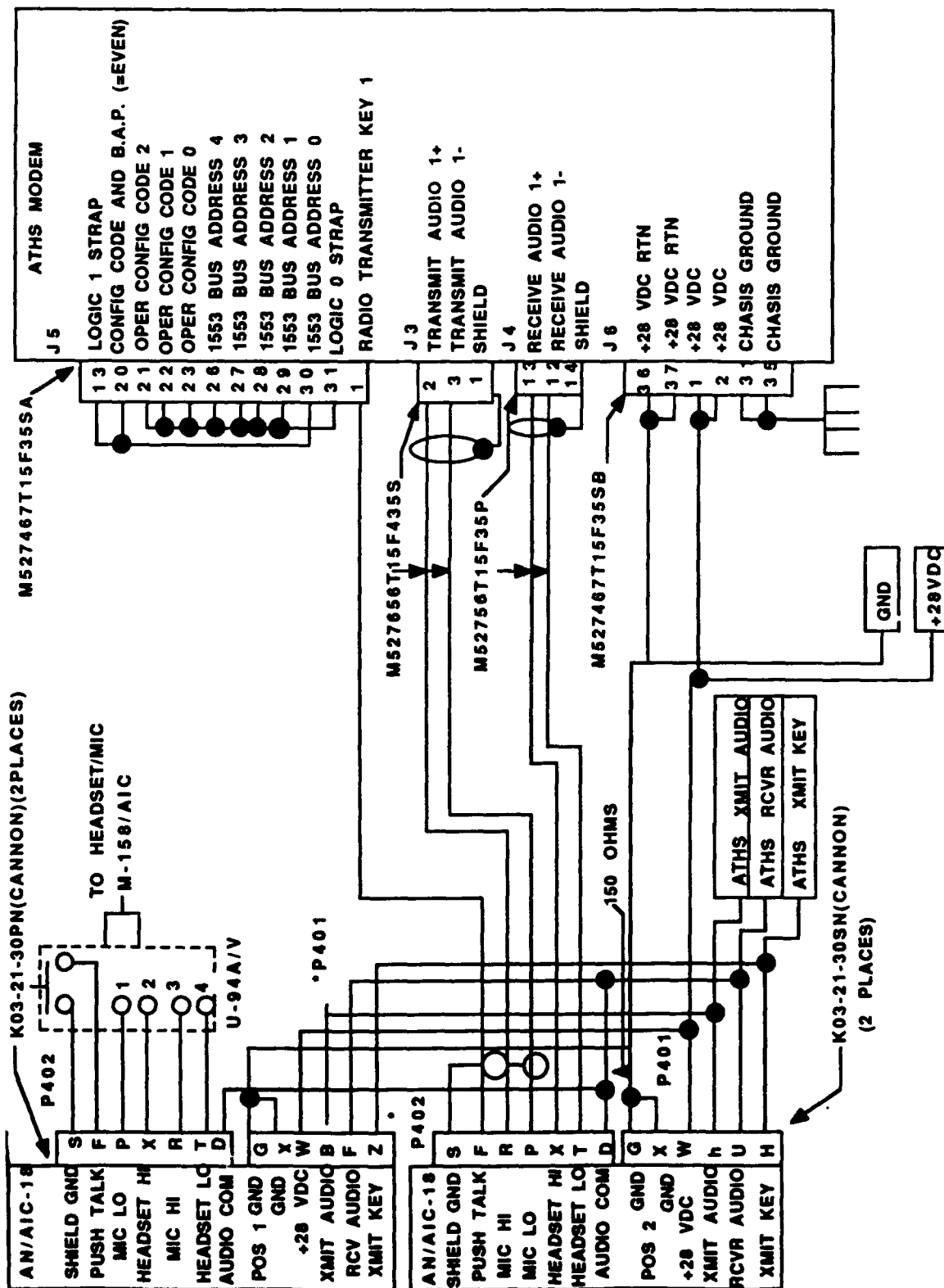


*** (Figure 2-5)**

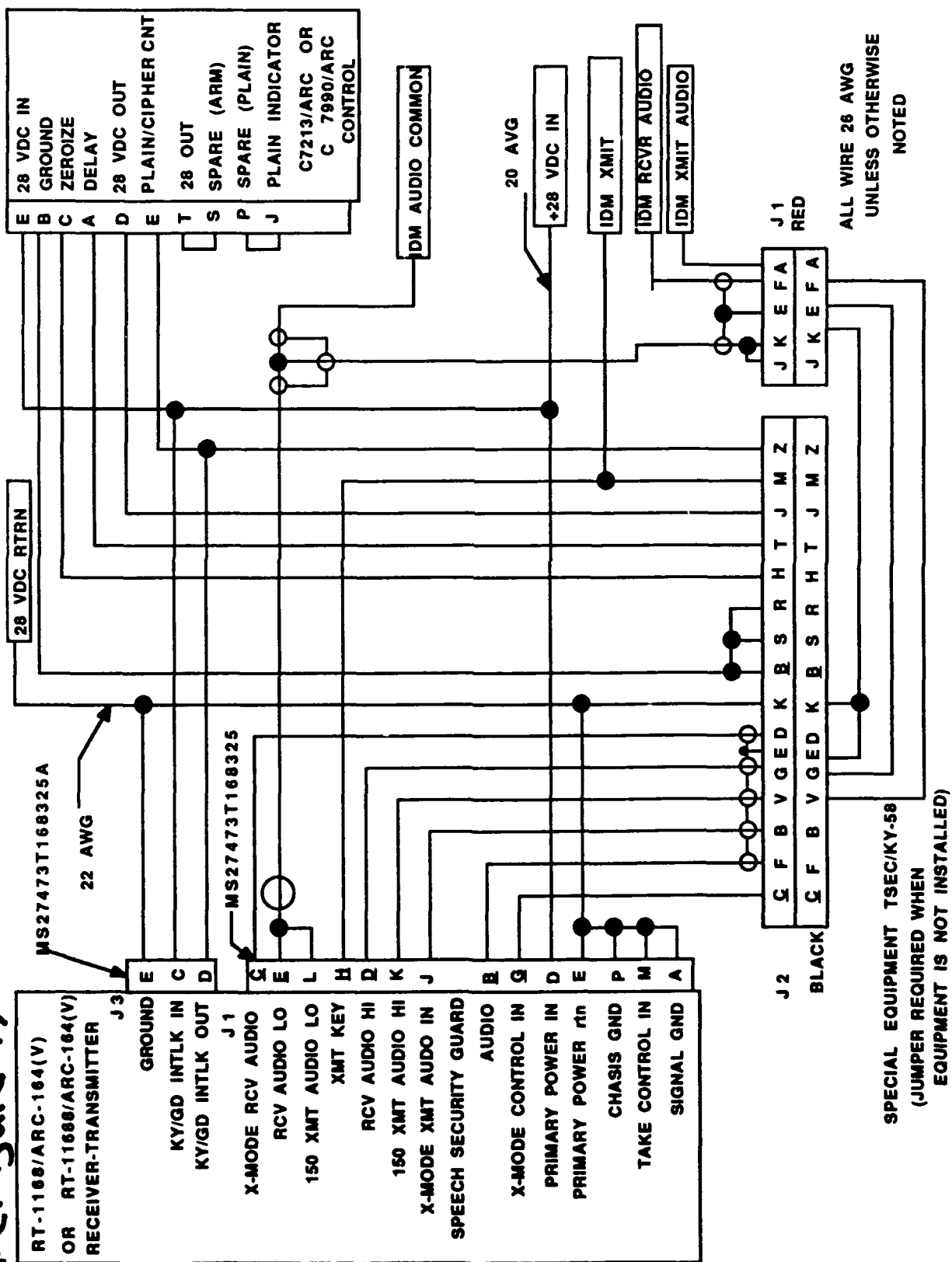
SPECIAL SPEECH EQPT TSEC/KY-58
JUMPER REQUIRED WHEN EQPT NOT
INSTALLED



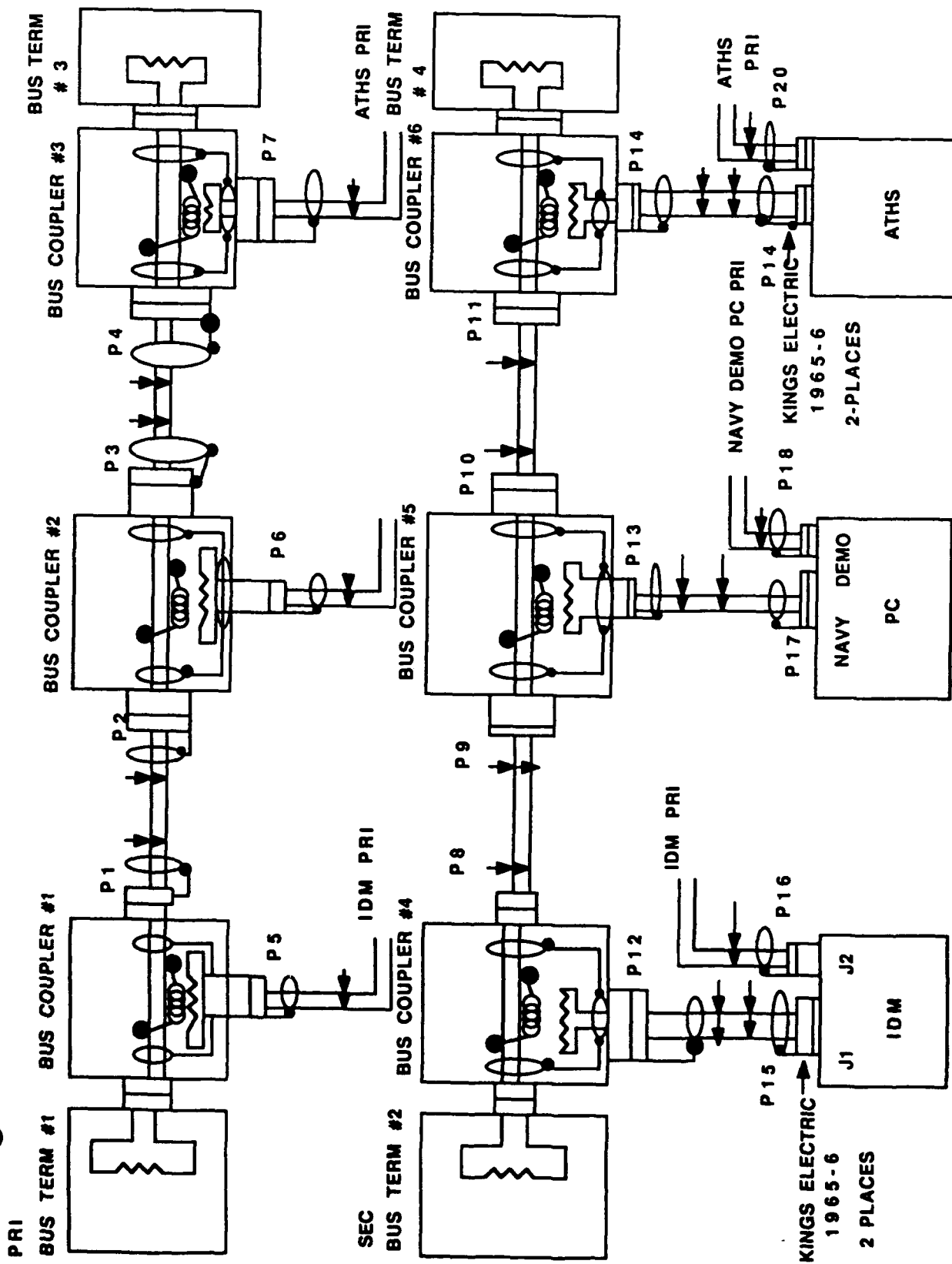
*(Figure 3)



***(Figure 4)**



*(Figure 5)



*(Figure 6)

LAB #1

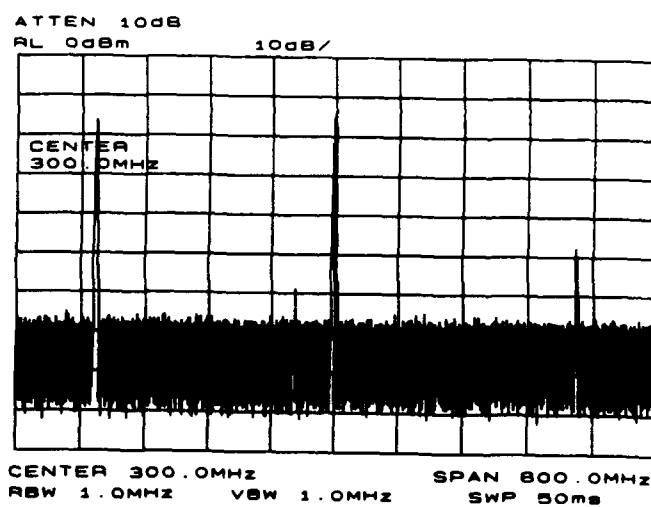
CREATE A SCENARIO IN UDOIT, ON EITHER SPECTRUM ANALIZER, TO SET:

- 1) CENTER FREQ. = 1,000 Khz
- 2) FREQ. SPAN = 10,000 Khz
- 3) SAVE SCENARIO UNDER (first name.sdf)
- 4) RUN THIS SCENARIO
- 5) CHECK RESULTS ON ANALIZER

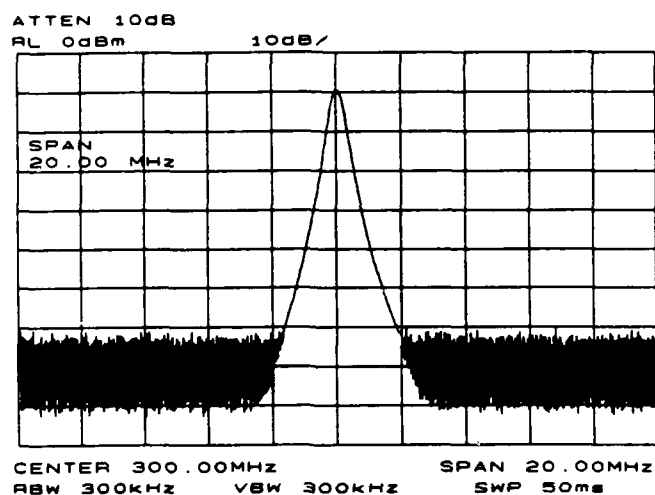
LAB #2

- 1) CREATE YOUR OWN SCENARIO
 - a) use at least three items to update in scenario
 - b) CENTER FREQ & FREQ SPAN can be used but will not be counted as one of the three items

*(Figure 7)

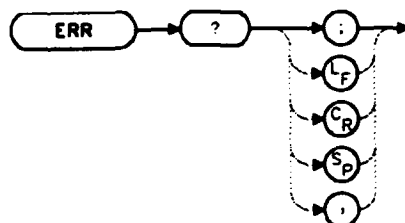


*(Figure 8)



* (Figure 9)

ERR Error



Description

The ERR command outputs a list of errors present. An error code of "0" means there are no errors present. For a list of error codes and descriptions, refer to Appendix C or the Installation and Verification Manual. Executing ERR clears all HP-IB errors. For best results, enter error data immediately after querying for errors. Each error code is three digits long.

*** (Figure 10)**

ACRONYMS

ICNIA	INTEGRATED COMMUNICATION NAVIGATION IDENTIFICATION AVIONICS
BIT	BUILT IN TEST
CNI	COMMUNICATION NAVIGATION IDENTIFICATION
MTBF	MEAN TIME BETWEEN FAILURE
LRU	LINE REPLACEABLE UNITS
LRM	LINE REPLACEABLE MODULES
AFSC	AIR FORCE SYSTEMS COMMAND
IESS	INTEGRATED ELECTROMAGNETIC SYSTEM SIMULATOR
DDF	DEVICE DESCRIPTOR FILE
ACSS	AVIONICS COMMUNICATION SYSTEMS SIMULATOR
UDOIT	USER DEFINED OPERATIONS AND INTERACTIVE TESTING
CSEL	COMMUNICATION SYSTEMS EVALUATION LABORATORY
DEC	DIGITAL EQUIPMENT CORPORATION
SASC	SYSTEMS AND APPLIED SCIENCES CORPORATION
HP-IB	HEWLETT - PACKARD - INTERFACE BUS

PT 2 Memo: Chaotic Functions Experiment

Christine Yoon

July 29, 1991

1 Introduction

This memo documents the results of PT 2 Task Order 3. The objective for this task was to decompose a series of logistic functions (part of a class of functions known as Chaotic) in order to demonstrate the generality of function decomposition in measuring complexity. It was our intent to discover whether or not a correlation existed between the DFC values for these functions and the Lyapunov Exponent for the portion of the Logistic Map from which these functions were taken. All the planned data we set out to be tested were generated and thus produced the results reported in this memo.

2 Summary of The Logistic Map

2.1 Introduction

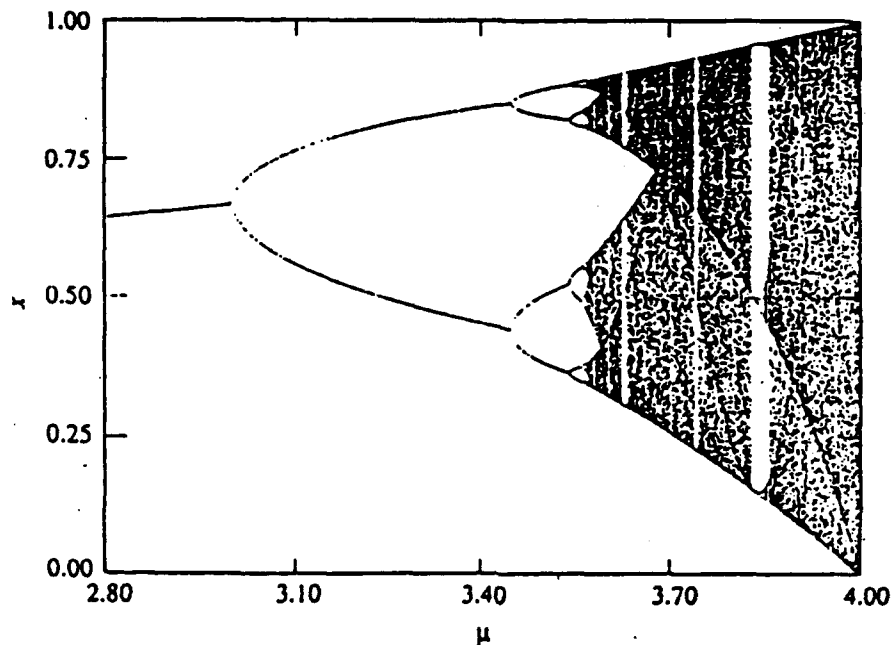
The Logistic Map is composed of a group of functions dependent on a constant μ which are of the form:

$$F_{\mu}(x) = \mu(x - x^2)$$

where μ is a positive real number less than 4.0. For any μ , an initial value for x can be chosen between zero and one. Each successive x is then dependent upon the preceeding x and is also bounded between zero and one.

For values of μ less than 3.0, the sequence converges to a single point. When $\mu = 3.0$, a bifurcation takes place and the sequence oscillates between two points. For values of μ between 3.44 and 3.48, two additional bifurcations take place and the sequence then endlessly visits four fixed points. As μ approaches 3.55, many additional bifurcations take place until the sequence becomes chaotic. Within the region of chaos, the fixed points become unstable and visitations among various small regions of numbers occur with no discernable pattern of repetition.

A plot of the two-dimensional 'logistic map' as it is commonly shown is reproduced in the figure below. This was taken from *Chaotic Dynamics of Nonlinear Systems* by S. Neil Rasband, p 21. The initial transient of 200 points has been discarded in each case to allow the sequence to converge to its fixed points in the non-chaotic region or to stabilize in the chaotic region.



3 The Experiment

A Turbo Pascal program was written by Lt. Tim Taylor on 20 JUN 91 (a copy of the code is included in appendix A) to create binary sequences, based on the logistic function for various values of μ so that we could use the AFD to decompose them. The program accepts input values for μ between 2.8 and 4.0. It also accepts initial x values between zero and one and it allows the user to decide how many initial transient points to discard. The program was designed to produce functions on eight variables, therefore, it produced a sequence of 256 real numbers between zero and one. In order to turn this sequence into a binary function, a one was assigned to any number in the sequence greater than .5 and a zero was assigned to any number .5 or less. For each number in the sequence, the program outputs the number of iterations, the value of x , the binary equivalent of the number of iterations (i.e. the binary input to the function) and the binary output. The program also produced a file containing the function in the ordinary format accepted by the Ada Function

Decomposition (AFD) program written by Chris Vogt in 1989.

To serve the purposes for our experiment, we always used .25 as our initial x and we consistently discarded the initial transient of 200 points to converge the output to its fixed points in the non-chaotic region and to allow it to stabilize in the chaotic region. Rasband has used this same approach, and we have experimental results (Appendix B) demonstrating that neither the final value of the function in the non-chaotic regions nor the essential complexity of the function in the chaotic regions is at all dependent upon the initial value chosen for x , given that the initial transients are discarded.

We used the Turbo Pascal program to generate approximately forty functions with values of μ as follows:

1. 3.5 to 3.9 at an interval of tenths and one point at 3.99
2. 3.61 to 3.85 at an interval of hundredths
3. 3.775 to 3.785 at an interval of thousandths (Lyapunov exponent high)
4. 3.825 to 3.835 at an interval of thousandths (Lyapunov exponent low)

The first group, at the interval of tenths, was chosen to give a broad look at the increasing complexity in the logistic map. The second group, at the interval of hundredths, was chosen to obtain a finer resolution in the chaotic region. The last two groups compare the relationship between complexity of the real functions of μ as measured by the Lyapunov exponent with the Decomposed Function Cardinality (DFC) of the binary functions which we produced for the same values of μ . The first of these two intervals was chosen in a region where the Lyapunov exponent was high, and the second was chosen where the Lyapunov exponent was low.

After generating the functions, we transferred them to the branch VAX 11/780 in order to decompose them using the AFD program. Each function was decomposed individually using version 2A. In cases where the DFC for a particular function seemed unusually high, the function was decomposed again, using version 4A. Three typical decompositions are included in Appendix D. All the decompositions were completed by 25 JUL 91.

4 The Results

The actual DFC values obtained for each region explored are shown graphically in the four charts included in appendix C. The first thing to notice is that decompositions were found for many of the sequences in the chaotic region, thus demonstrating the generality of DFC as a measure of complexity. It's visually apparent that the complexity of the Logistic Map increases as one

moves toward the right, and chart C.1 shows an overall steady increase in DFC with the increase in μ .

Chart C.2 begins to give us an indication of the existence of some relatively stable regions within the chaotic region. Although there is an overall steady increase in DFC with increasing μ , there are some values of μ (notably 3.63, 3.74 and 3.83) where the DFC suddenly drops. A quick glance at the Logistic Map will show that these are the same places where the functions become visibly less chaotic.

Rasband states that the Lyapunov exponent is a good indicator of chaos. Our experiments have shown that there does indeed appear to be a strong correlation between the DFC values and the Lyapunov exponent. Where the DFC is high, the Lyapunov exponent is high and where the DFC is low, the Lyapunov exponent is low. In chart C.3 where the Lyapunov exponent was relatively stable and high (see chart C.5) the average DFC over all the functions in the region was 224 (87.5 %DFC) whereas in chart C.4, where the Lyapunov exponent was relatively low and showed a sharp dip, the average DFC was 118 (46.1 %DFC) and the graph also shows a dip to a DFC of 0 when $\mu = 3.832$. After decomposing all forty-nine functions which comprised the experiment, we calculated an approximation of the Lyapunov Exponent for each value of μ and computed the overall correlation. It was found to be .904. Further statistical analysis of the relationship between DFC and the Lyapunov exponent as well as a table of all the computed values is shown in Appendix E.

The obvious and significant conclusion to be drawn from the experiment is that DFC is a robust measure of complexity for chaotic function as well as non-chaotic functions.

5 Notes on AFD Version 4A

The initial experiments with the logistic map which were undertaken by T. Ross and J. Wolper, showed the same overall trends in DFC which we have reported here for values of μ between 3.6 and 3.8 as calculated by version 2A. However, when some of the functions with higher DFC's were decomposed again using version 4A, the new DFC's which were found were *significantly* lower. Many were more than 25% lower than those found by version 2A. This sparked some interest at the time, as all the previous data gathered by the phenomonology study had shown that the difference in DFC from the slowest version (version 3) to the fastest version (version 2A) averaged on the order of 2% to 5%. Our initial assumption was that there was something unique about the chaotic functions which caused their decompositions to be significantly "trickier" than most of the 800 or so functions which we had decomposed previously.

However, as we got farther into the experiment, we noticed that many of

the functions in the chaotic region were decomposing to a total subfunction cardinality of 224 when decomposed on version 2A but had a DFC of 160 when decomposed on version 4A. All of these decompositions had similar structures and in each case a look at the V4A decompositions readily showed that version 4A had discovered the same decomposition as that found by V2A, but had failed to record one of the subfunctions (of cardinality 64) in the output, thus giving rise to an erroneous DFC. Once we found the first error, it was easy to produce additional examples, and this led to a reexamination and correction of the data in the tech report.

None of the changes were very significant, but the discovery of the error in V4A was important for several reasons. Most importantly, if the DFC's produced by V4A were correct, it would have thrown some doubt on our contention that the decompositions produced by V2A were near optimal, which might have made finding a good heuristic for future decomposition algorithms more complicated.

```

program logistics;

uses
  crt;

type
  bin = (zero, one, dont_care);

var
  logistic : array[0..255] of bin;
  i,iin : integer;
  r,x : real;
  input,output : integer;
  dln : longint;
  bstring : string[8];
  file_name, function_name : string[10];

  outfile : text;

procedure gen_func;

begin
  x := r*x*(1-x);
end;

procedure get_bin;

var
  count,k : integer;
  bin : array[0..7] of shortint;
  j : shortint;
  dummy : string[11];

begin
  k := i;
  count := 128;
  bstring := '';
  for j := 0 to 7 do
    begin
      if k >= count then
        begin
          bin[j] := 1;
          k := k - count;
        end
      else
        bin[j] := 0;
        count := count div 2;
        str(bin[j], dummy);
        bstring := bstring + dummy;
      end;
    end;
end;

```

```
procedure get_values;
```

```
begin
  clrscr;
  writeln('Please enter a number for r (between 2.8 and 4.0)');
  readln(r);
  writeln('Please enter a delay loop number');
  readln(dln);
  repeat
    writeln('Please enter an initial x value (between 0 and 1)');
    readln(x);
  until (x>0.0) and (x<1.0);
  writeln('Please enter an initial iteration number');
  readln(iin);
  writeln('Please enter a file name');
  readln(file_name);
  writeln('Please enter a function name');
  readln(function_name);
end;
```

```
procedure delay;
```

```
var
  i : longint;

begin
  for i := 1 to dln do;
end;
```

```
procedure write_file;
```

```
var
  i,k : integer;

begin
  assign(outfile,file_name);
  rewrite(outfile);
  k := 0;
  for i := 0 to 255 do
    if (logistic[i] = one) then
      k := k+1;
  writeln(outfile,'1');
  writeln(outfile,function_name);
  writeln(outfile,'8');
  writeln(outfile,'1');
  writeln(outfile,k);
  for i := 0 to 255 do
    if (logistic[i] = one) then
      writeln(outfile,i);
  writeln(outfile,'2');
  close(outfile);
end;
```

```

begin
  get_values;
  clrscr;
  writeln('iterations    mu      x[i]    input    output');
  for i := 1 to iin do
    gen_func;
    for i := 0 to 255 do
      begin
        get_bin;
        delay;
        if x > 0.5 then
          begin
            output := 1;
            logistic[i] := one;
          end
        else
          begin
            output := 0;
            logistic[i] := zero;
          end;
        writeln(i:9, ' ', r:7:3, ' ', x:7:5, ' ', bstring, ' ', output);
        gen_func;
      end;
    write_file;
  end.

```


It can be easily demonstrated that logistic functions which converge to a single point or a finite and small number of points will exhibit this convergence regardless of the initial value chosen for x . However, demonstrating that a chaotic logistic function retains the same degree of complexity, independent of the initial x , is a little more difficult. When the value of μ is sufficiently great that the function will not converge to any finite sequence, different initial x 's will always generate completely different infinite sequences. We know that the Lyapunov exponent, which is a common measure of complexity in chaotic systems, is completely independent of the initial x , so we felt it would be good to check whether or not the DFC for any given μ was dependent on the initial x .

We chose $\mu = 3.8$ and generated four sequences, using initial x 's of 0.25, 0.4, 0.6 and 0.8 and eliminating the first 200 elements. The functions were named 'log38', 'log38a', 'log38b' and 'log38c' respectively. The decompositions are shown on the following pages. Although the functions are clearly different and chaotic, it is also clear that they decompose to the same cost and in exactly the same way.

```

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the functi
Enter decimal equivalent of binary input that has a true value: Enter decimal eq
A 3

```

number of variables:

lost - 256

```
Cost =      64
           1      2      3      4      5      6      7
01001101010001010111000010101000010010011110011100100010101100011
```

CPU Time:
99.33

SPECIFY FUNCTION TABLE VALUES:

30-10

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the function
Enter decimal equivalent of binary input that has a true value: Enter decimal eq

2A 3

Date:

7

31

1991

Function name:

log38b

Number of variables:

8

1	2	3	4	5	6
11111011010111101010110111101010110111101011110101110110					
1010110111101101111011101111111011101010101111110111010110111					
1111011011101101111010111101101110101110101111010101010101					
110101111010111011101101011110111010111101011101010101110110111					

Cost = 256

Better decomp found

7	8	10	11	12	9
1101110010110111101101111100110					

Cost = 32

1	2	3	4	5	6	10
0000110111111111101011001100001010100010101101110100000101001						

Cost = 64

1	2	3	4	5	6	11
0011001000010011001001100000110101010100001010010100000111010001						

Cost = 64

1	2	3	4	5	6	12
01010110111001010111000010001110010100111011101010111101110110						

Cost = 64

Decomposition cost: 224

Node Information:

0	1	2	3	4	5	6
1	4	0	0	0	0	0

CPU Time:

90.97

CPU Time:

100.41

Node Information:

0	1	2	3	4	5	6
1	4	0	0	0	0	0

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Thank you for using the PBML function Decomposer

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the function have?
Enter decimal equivalent of binary input that has a true value: Enter decimal: eq
2A 3
Date:

7 31 1991

Function name:

log38c

Number of variables:

8
1 2 3 4 5 6
101011011101110101011101111101101111101111101101010110111101101
1111010101011011101011110111110111101110101010111101010110111
1011010111011101010110111110101011111111010110111101011111111
1111110111110101011101111010101010111011110111011101111011011

Cost = 256

Better decomp found

7 8 10 11 12 9
11011010011101111001111101111100

Cost = 32
1 2 3 4 5 6 10
1000000110110010000000011000101110000110001110000100010011111101

Cost = 64
1 2 3 4 5 6 10
10001010010100101100001000001000001001010111000111110011110011000

Cost = 64
1 2 3 4 5 6 10
111011011000101100110100110100100110000110010111011001010000010

Cost = 64
Decomposition cost: 224
Node Information:

0 1 2 3 4 5 6
1 4 0 0 0 0 0

CPU Time:
88.71

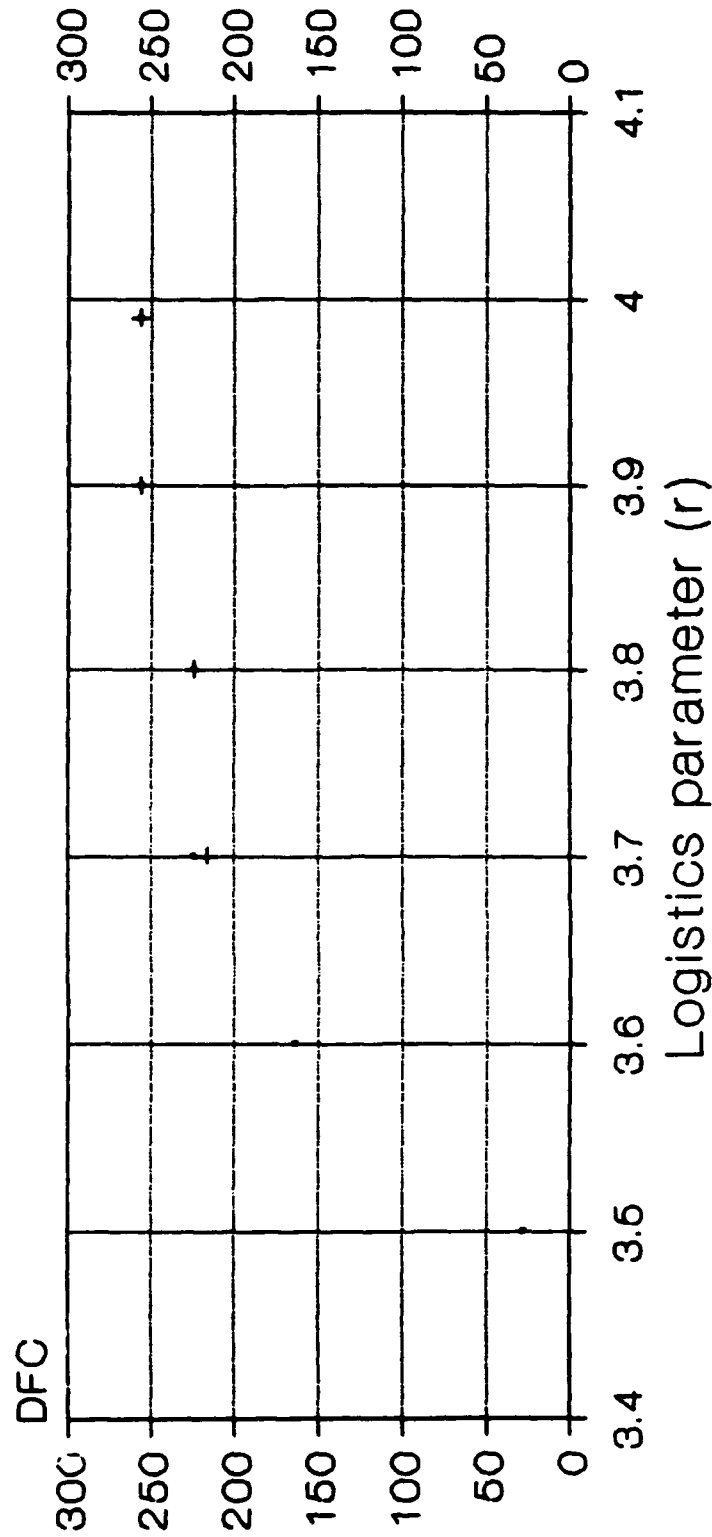
CPU Time:
97.82

Node Information:
0 1 2 3 4 5 6
1 4 0 0 0 0 0

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Thank you for using the PBML function Decomposer

Logistic Functions 2A and 4A



N = 8

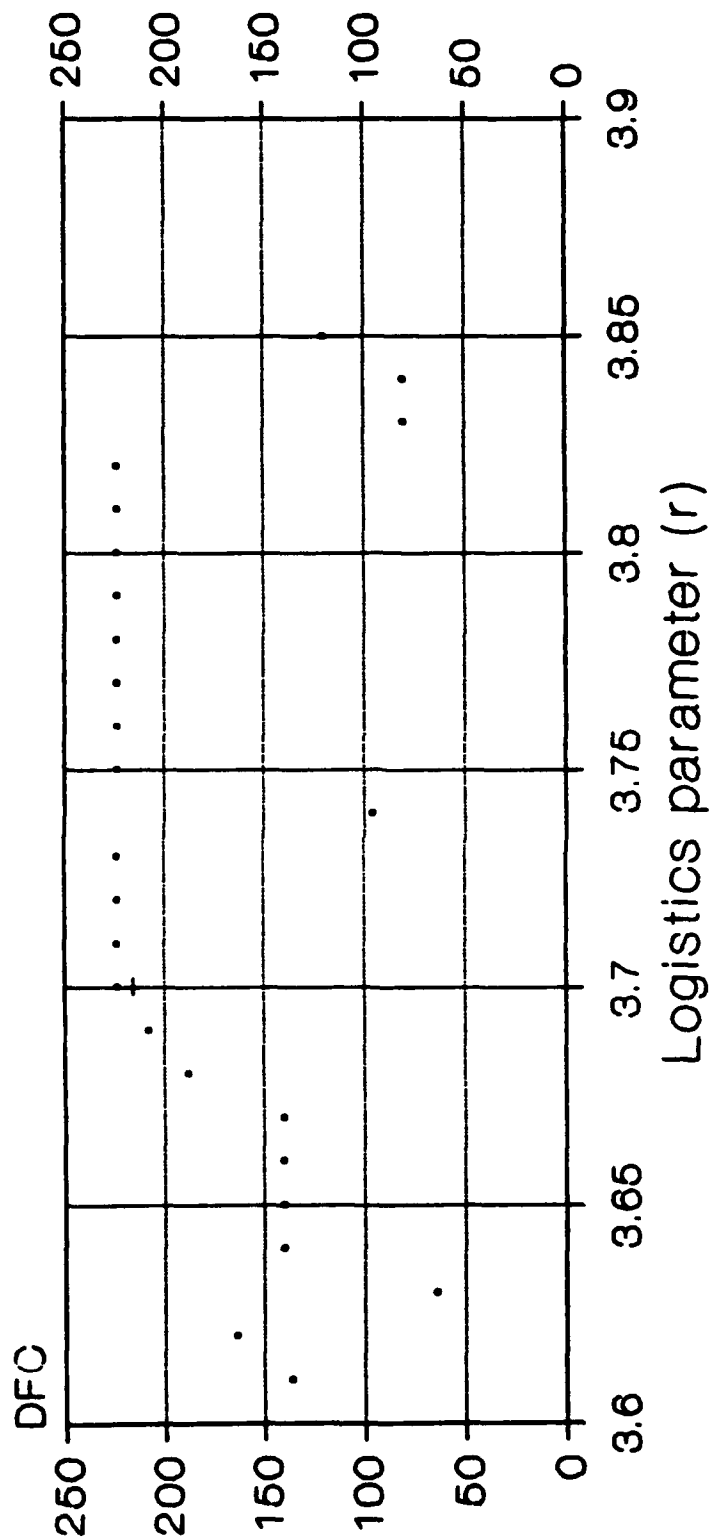
• V2A + V4A

Appendix C

7/17/91

Logistic Functions

2A and 4A



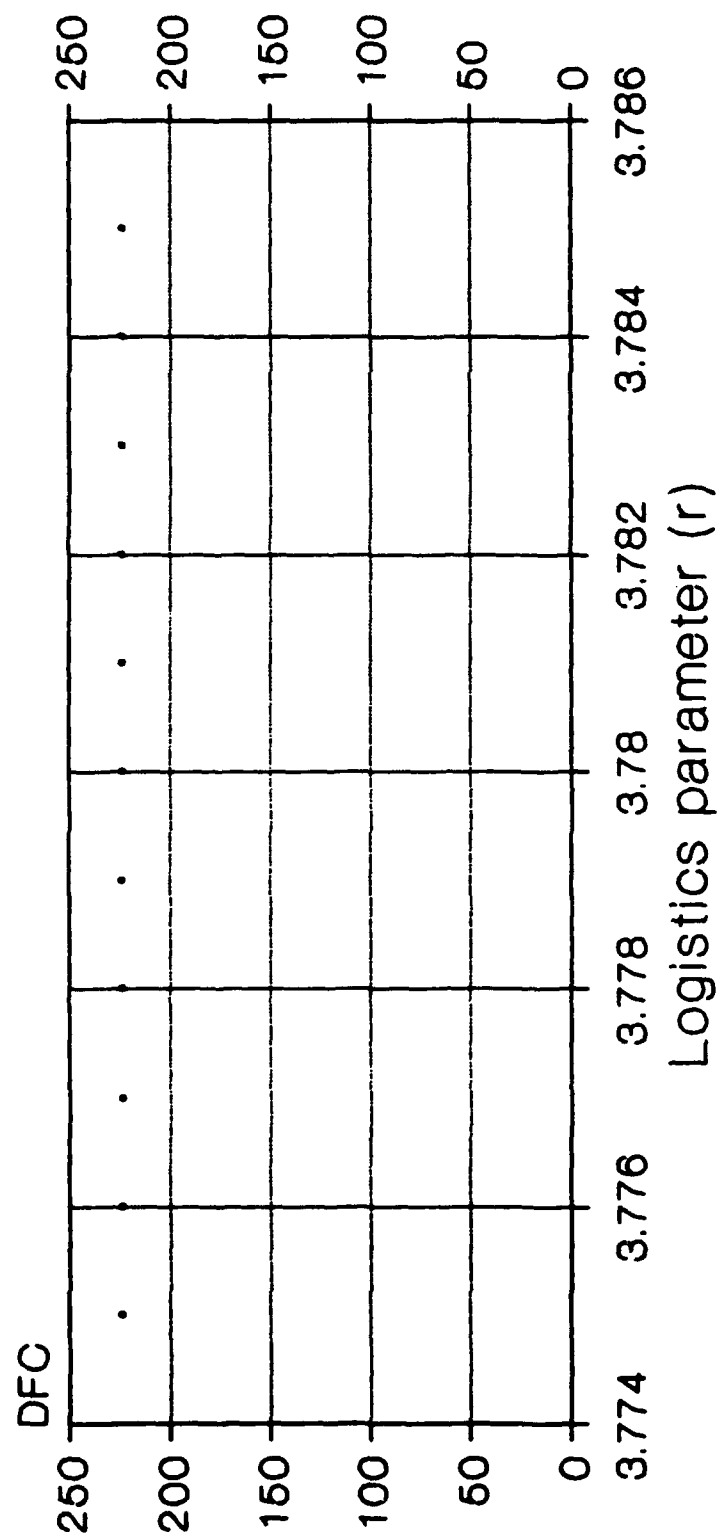
N = 8

• V2A + V4A

7/17/91

Logistic Functions

2A



N = 8

• V2A

7/17/91

Appendix D

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the functi
Enter decimal equivalent of binary input that has a true value: Enter decimal eq
2A 3

Date: 7 3 1991

Function name:

log35

Number of variables:

8	7	6	5	4	3	2	1
0110111011101110111011101110111011101110111011101110111011101110							
1110111011101110111011101110111011101110111011101110111011101110							
1110111011101110111011101110111011101110111011101110111011101110							
1110111011101110111011101110111011101110111011101110111011101110							

Cost = 256

Better decomp found

7	8	14	9
01111100			
Cost = 12	8	13	14
0111			
Cost = 10	4	11	13
0111			
Cost = 5	4	6	12
0111			
Cost = 3	4	4	11
0111			
Cost = 1	4	2	10
0111			
Cost = 4			

Decomposition cost: 28

Node Information:

0	1	2	3	4	5	6
1	2	20	120	360	360	1

CPU Time:
234.44

CPU Time:
3442.82

Node Information:

0	1	2	3	4	5	6
1	30	300	1800	5400	5400	1

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Thank you for using the PBML function Decomposer


```
Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the func
Enter decimal equivalent of binary input that has a true value: Enter decimal eq
```

7 9 1991

1093833

8

[illegible]

Better decomp found

Cost =

00011000

Cost =

30100100

— — — — —

Cost =

30100100

01001001

Cost -

Node Info

CPU Time

CPU TIME:
1627.63

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Name of function: How many input variables does the functi
Enter decimal equivalent of binary input that has a true value: Enter decimal eq
2A 3

Date: 7 3 1991

Function name:

log38

Number of variables:

8						
1	2	3	4	5	6	
1101010101011011101101110110111011101110111011101111111						
11101111101110110101111101010110111110110111011111101101						
111101010111101101011110110111010101010101111010101010101						
11101010101110111101011111011101111011011110111011011010						

Cost = 256

Better decomp found

7	8	10	11	12	9
1010111011010111001111111110100					

Cost = 32

1	2	3	4	5	6	10
0000000000001101110011100100111010001100010011001100101000000001						

Cost = 64

1	2	3	4	5	6	10
0001111111110010100000011100101001100011001010101011110111010010						

Cost = 64

1	2	3	4	5	6	10
0110010111110101010001001000110010100101100100010000001101000000						

Cost = 64

Decomposition cost: 224

Node Information:

0	1	2	3	4	5	10
1	4	0	0	0	0	0

CPU Time:

90.11

CPU Time:

99.27

Node Information:

0	1	2	3	4	5	10
1	4	0	0	0	0	0

SPECIFY FUNCTION TABLE VALUES:

Enter 0 to input from a file, 1 from terminal,
or 2 to QUIT program: Thank you for using the PBNL function Decomposer

Appendix E

μ	DFC	L.E.
3.500	28	-0.86556
3.600	164	0.18119
3.610	136	0.19549
3.620	144	0.19971
3.630	64	-0.01368
3.640	140	0.22836
3.650	140	0.24818
3.660	140	0.28414
3.670	140	0.30697
3.680	188	0.34556
3.690	206	0.35006
3.700	224	0.34954
3.710	224	0.36098
3.720	224	0.36320
3.730	224	0.38370
3.740	96	-0.10409
3.750	224	0.35657
3.760	224	0.38296
3.770	224	0.40332
3.775	224	0.39056
3.776	224	0.37593
3.777	224	0.39225
3.778	224	0.38625
3.779	224	0.38448
3.780	224	0.39852
3.781	224	0.40816
3.782	224	0.39611
3.783	224	0.41319
3.784	224	0.41519
3.785	224	0.41704
3.790	224	0.42624
3.800	224	0.43586
3.810	224	0.42451
3.820	224	0.43253
3.825	224	0.42416
3.826	224	0.39960
3.827	224	0.35102
3.828	196	0.31287
3.829	80	-0.17031
3.830	80	-0.36820
3.831	90	-0.65557
3.832	0	-1.32485
3.833	80	-0.61735
3.834	80	-0.42281
3.835	80	-0.30847
3.840	80	-0.04465
3.850	120	0.00817
3.900	256	0.48670
3.990	256	0.63684

```

ccoeff
enter input filename
a:le_dfc.dat
read: 2.8000000000E+01-8.6556000000E-01
read: 1.6400000000E+02 1.8119000000E-01
read: 1.3600000000E+02 1.9549000000E-01
read: 1.6400000000E+02 1.9971000000E-01
read: 6.4000000000E+01-1.3680000000E-02
read: 1.4000000000E+02 2.2836000000E-01
read: 1.4000000000E+02 2.4818000000E-01
read: 1.4000000000E+02 2.8414000000E-01
read: 1.4000000000E+02 3.0697000000E-01
read: 1.8800000000E+02 3.4556000000E-01
read: 2.0800000000E+02 3.5006000000E-01
read: 2.2400000000E+02 3.4954000000E-01
read: 2.2400000000E+02 3.6098000000E-01
read: 2.2400000000E+02 3.6320000000E-01
read: 2.2400000000E+02 3.8370000000E-01
read: 9.6000000000E+01-1.0409000000E-01
read: 2.2400000000E+02 3.5657000000E-01
read: 2.2400000000E+02 3.8296000000E-01
read: 2.2400000000E+02 4.0372000000E-01
read: 2.2400000000E+02 3.3056000000E-01
read: 2.2400000000E+02 3.7593000000E-01
read: 2.2400000000E+02 3.9225000000E-01
read: 2.2400000000E+02 3.8625000000E-01
read: 2.2400000000E+02 3.8448000000E-01
read: 2.2400000000E+02 3.9852000000E-01
read: 2.2400000000E+02 4.0816000000E-01
read: 2.2400000000E+02 3.9611000000E-01
read: 2.2400000000E+02 4.1319000000E-01
read: 2.2400000000E+02 4.1519000000E-01
read: 2.2400000000E+02 4.1704000000E-01
read: 2.2400000000E+02 4.2624000000E-01
read: 2.2400000000E+02 4.3586000000E-01
read: 2.2400000000E+02 4.2451000000E-01
read: 2.2400000000E+02 4.3253000000E-01
read: 2.2400000000E+02 4.2416000000E-01
read: 2.2400000000E+02 3.9960000000E-01
read: 2.2400000000E+02 3.5102000000E-01
read: 1.9600000000E+02 3.1287000000E-01
read: 8.0000000000E+01-1.7031000000E-01
read: 8.0000000000E+01-3.6820000000E-01
read: 8.0000000000E+01-6.5557000000E-01
read: 0.0000000000E+00-1.3248500000E+00
read: 8.0000000000E+01-6.1735000000E-01
read: 8.0000000000E+01-4.2281000000E-01
read: 8.0000000000E+01-3.0897000000E-01
read: 8.0000000000E+01-4.4650000000E-02
read: 1.2000000000E+02 8.1700000000E-03
read: 2.5600000000E+02 4.8670000000E-01
read: 2.5600000000E+02 6.3684000000E-01
r= 9.0432065263E-01
n=49 x_avg= 1.7542857143E+02 y_avg= 1.7755244898E-01
s_x= 4.4591020408E+03 s_y= 1.5255236545E-01 s_xy= 2.3586072420E+01

```

High School Apprenticeship Program

Final Paper

Mark Buxton (610)

August 23, 1991

I would like to thank Mr. Davis, my mentor, for his great expenditure of time and wise advice. Without him, this summer would have been awfully dull and my eyes never would have been opened to the world of engineering. I would also like to thank everyone else for their help an for putting up with my hacking. I am sure the Vax will compute a sigh of relief when I leave.

:

Final Report for the High School Apprenticeship Program

Monte Carlo Simulation was performed on various electron beam resists using the methods of Hawryluk et al. [2], Bishop [4], and Murata et al. [3]. Simulation was conducted on both bulk and layered resists and results were evaluated using RS1.

The Classical laws of optics were used to characterize the diffractive interference of a sub-micron grating upon 680nm light.

This page outlines the research activities completed during the time span June 17 to August 26, 1991. The research took place at the Wright-Patterson Air Force Base in the Electronics Research Division.

The first main project undertaken in the ten week apprenticeship was the simulation of an electron beam in its interaction with various resists. The resists studied include PMMA (Methyl ester of 2-Methylpropenoic acid) and MAA (2-Methylpropenoic acid). The goals of this simulation were

1. to provide a model for the 2-dimensional exposure of the resist as a function of beam energy and resist parameters,
2. to write a program to simulate the exposure of multi-layer systems under various conditions,
3. to present results in a format accessible for further use in development simulation and for presentation.

The first goal was met in the method outlined in the following section, "Monte Carlo Simulation of 50-keV Electrons in Various Resists." The methods of various authors [1-6] were used to construct an exposure model

based on the MKS system (as opposed to the CGS system commonly employed). The second goal was met through programs written using Vax-Pascal and Borland Pascal. The programs allow easy manipulation of resist characteristics and initial beam characteristics. The third goal was met through the use of the RS1 graphing program to generate contour plot graphs.

The second major research project involved the evaluation of the effect of shining 680nm light through a sub-micron diffraction grating. The goal of this project was to simulate, using the classical laws of optics, the diffractive interference produced by the light on a screen a given distance from the grating. This goal was met through the construction of Pascal and Microsoft Excel programs to generate appropriate data, and was graphed in RS1. The method of evaluation and results can be found in the following section, "Effect of a Sub-Micron Grating on Coherent Light."

Two smaller projects were completed. The first was the labeling of laboratory chemicals using a labeling program constructed by Matt Elwood. The second was the testing of two samples using the method of X-ray backscattering spectroscopy. Both are outlined as the final sections of this paper.

Monte Carlo Simulation of 50-keV Electrons in Various Resists

Wright-Patterson Air Force Base is currently committed to research in the fields of microelectronics. This research includes designing new devices, testing manufacturing schemes and acquiring working processes, testing finished devices, and working with the materials from which the devices are manufactured. One of the goals of this research is to provide information in high-risk areas or areas with only long-term economic benefit. One area the Electronics Research Division is currently exploring is the construction of ever smaller devices by using electron beam technology.

The manufacture of semiconductor devices can be broken down into four main steps,

1. Evaluation of required performance characteristics,
2. Design of devices,
3. Manufacture of devices,
4. Testing of the finished products.

Step three, the manufacture of microelectronic devices can, in the case of semiconductor-based devices, be further broken down into

1. Wafer preparation (selection, cleaning, doping),
2. Application of an energy sensitive "resist",
3. Exposure and development of the resist (including etching, metallization, doping, and molecular beam epitaxy or other deposition techniques),

4. Removal of resist/repeat of above steps.

The exposure of the energy resists is accomplished by one or both of two methods, using light as the energy transfer device or using beams of electrons. In the region of sub-micron and nanometer device fabrication, electron beam lithography (a method of patterning the resist with the electron beam) provides a significant advantage to optical methods. Electron beam lithography theoretically can produce higher resolution of exposure and therefore smaller devices than the equivalent optical lithography techniques due to the small wavelength of the electron.

Electron beam lithography is, however, plagued with technical and material problems which may offset its usefulness in certain applications. The actual exposure of a device is confined to a single beam of electrons, and for all the speed at which the beam is able to scan the surface of the resist, the rate of device production with this technique will never equal that of optical lithography. Determination of exposure and development times with the electron beam system is complicated by the lack of a good model predicting how the resist will react to combinations of exposure and development. The nature of the electron beam itself is a cause of problems because the beam undergoes scattering as soon as it contacts a material surface. The surface reacts to the electrons by becoming charged, leading to beam drift. These effects have the

result of making the prediction of the actual feature parameters more of an art than a science.

One tool which has been used with some success [1-4,6] in the past few decades to predict the interaction of electron beams with matter is the monte carlo simulation. This method uses a computer to simulate individual electrons and their semi-random walk through the target using theories of atomic and molecular interaction. Because the accuracy of the simulation is proportional to the square-root of the number of tested electrons, a large number of test particles is required for useful results.

Monte Carlo Simulation and Results

When an electron enters a material it encounters forces from each of the constituent atoms of that material. A commonly used method of simulating electrons in materials is to take the change in momentum resultant from these forces as occurring only in a specified region, the "range" of the force. One frequently used method of defining the probability of the incident electron encountering one of these forces is by assigning a two-dimensional "cross section," σ , to each atomic species [5]. Thus, the probability of hitting an atom of that species in a given distance of travel in the medium corresponds directly to the magnitude of that cross-section.

The forces which define the cross section of the atoms can be taken to be Coulomb forces acting on the electrons at a distance r ,

$$V(r) := \frac{Ze^2}{r} \cdot \exp(-\lambda \cdot r) \quad (1)$$

with a hit atom of atomic number Z and a fundamental charge e . The exponential part on the right hand side of eq. (1) is to account for a discrepancy of the theory for large values of Z [1]. Nigam [1] defines the amplitude of the scattered particles as a function of the angle of scattering θ ,
(3)

$$f(\theta) := Z \cdot e^2 \cdot \frac{m}{4 \cdot E \cdot \left[\sin^2 \left(\frac{\theta}{2} \right) + \frac{\lambda^2}{h^2} \right]} \quad (2)$$

for an electron of mass m with h as Planck's constant, which yields the differential scattering cross section in two dimensions, also called the screened Rutherford cross section, which is given by Hawryluk et al. [2] as

$$\frac{d\sigma}{d\Omega} := \frac{Z \cdot (Z + 1) \cdot e^4}{16 \cdot E^2 \cdot \left[\sin^2 \left(\frac{\theta}{2} \right) + \alpha^2 \right]^2}$$

where α has replaced $\lambda/(h \cdot 1.414)$ and $Z \cdot (Z + 1)$ has been replaced Z^2 by Nigam to better fit results. Eq. (3) can be rewritten to eliminate the half angle,

$$\frac{d\sigma}{d\Omega} := \frac{Z \cdot (Z + 1) \cdot e^4}{16 \cdot E^2 \cdot [1 + 2 \cdot \alpha^2 - \cos(\theta)]^2} \quad (4)$$

The cross section σ can be obtained by integrating eq. (4) over the solid angle Ω and is given by Murata [3] as

$$\sigma := \frac{Z \cdot (Z + 1) \cdot e^4 \cdot \pi}{4 \cdot E^2 \cdot \alpha^2 \cdot [\alpha^2 + 1]} \quad (5)$$

The total distance traveled is related inversely to both the size of this scattering cross section and the concentration of scattering cross sections per unit volume. If we take the number of atoms of species i per unit volume N with cross section σ , the mean free path S will be given by

$$S := \frac{1}{\sum_i N_i \cdot \sigma_i} \quad (6)$$

Hawryluk suggested to predict a step length L so that these sub-steps form a Normal distribution of length,

$$L := -S \cdot \ln(0 < \text{random} < 1).$$

Thus the step length resulting from each electron-atom collision may be found with the appropriate value of α . To complete the simulation, the path must also be found by acquiring the angle of deflection θ . The radial deflection ϕ is ignored as the simulation is only two dimensional, but will be accounted for later. The energy loss per collision is the remaining value that must be computed to give the simulation meaning. One method of finding the differential energy loss per distance is that used by Bishop [4], the continuous slowing down approximation of Bethe. Bishop gives a representation,

$$\frac{dE}{dS} := \frac{- \left[1.984 \cdot 10^{-25} \cdot \rho \right]}{Z \cdot E} \cdot \ln \left[\frac{E}{J} \cdot 1.166 \right] \quad \frac{J}{m} \quad (7)$$

with atomic number Z of the struck atom, E as the energy in Joules, J as the mean ionization potential of the atom in Joules, and density ρ in g/ccm. Thus, the energy lost at each interaction is eq. (5) times the change in distance S .

The remaining variable is the angle of scattering θ . Hawryluk [2] gives a way of computing θ from α , namely,

$$\cos(\theta) := \frac{\text{Random} \cdot \left[1 + 2 \cdot \alpha^2 \right] - \alpha^2}{\text{Random} + \alpha^2} \quad (8)$$

The simulation was performed on a model of three layers of different types of resists, as pictured. The substrate was included to allow some of the normally scattered electrons to return into the resist. Simulation was also done on solid blocks of resists. Once the data for n electrons had been computed using the step length, scattering angle, and energy loss given previously, the results were normalized for exposures per cell. The total energy deposited per annulus of area

$$A_{\text{annulus}} := \pi \cdot \delta^2 \cdot (2 \cdot r - \delta) \quad ,$$

radius r , and square cross section $\delta \times \delta$ is merely the energy found in the simulation. The energy deposited

per cell of area $\delta x \delta y$ per electron is then

$$E_{\text{electron}} := \frac{\delta}{\pi \cdot (2 \cdot r - \delta)} \cdot \frac{E}{n}$$

The total number of electrons in an actual exposure of one point is given by

$$n_{\text{exposure}} := \frac{I}{T} \cdot 6.25 \times 10^{18}$$

for a shot with dwell time T in seconds and a beam current I in amperes. The energy deposited per cell in joules per shot then becomes

$$E_{\text{cell}} := \frac{\delta \cdot E \cdot I \cdot 6.25 \times 10^{18}}{\pi \cdot (2 \cdot r - \delta) \cdot n \cdot T}$$

The results of this simulation are given in the accompanying graphs. The exposure assumes the classic "teardrop" shape associated with beam spreading. In the simulation, a Gaussian electron beam was modeled and used for the initial radial distribution. The value used for α is that given by Hawryluk,

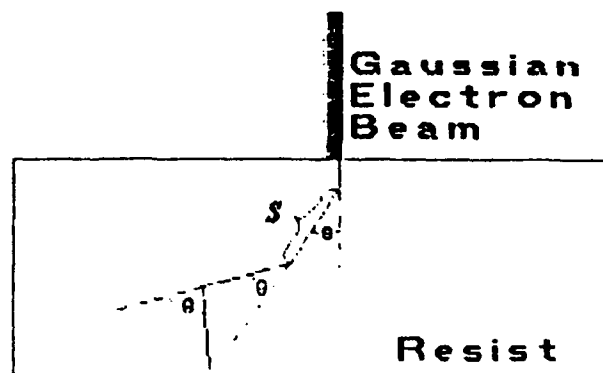
$$\alpha := 9.32 \times 10^{-10} \frac{Z^{0.33}}{\sqrt{E}}$$

with E in Joules.

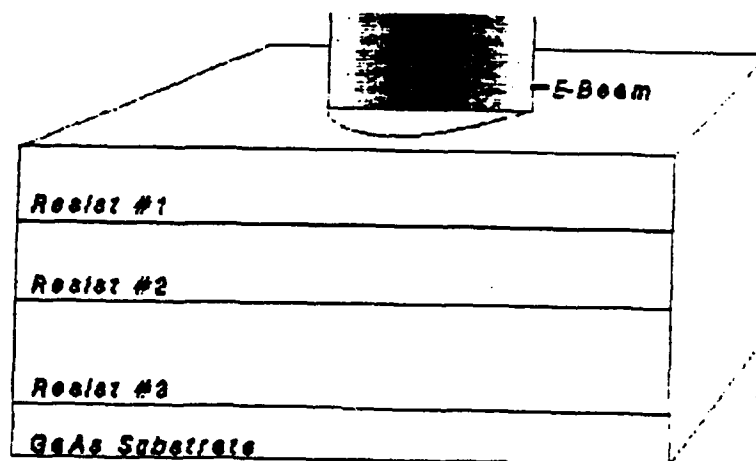
The atom with which the electron collided was given by Hawryluk as the probability:

$$P_i := \frac{\sigma_i \cdot n_i}{\sum_i [\sigma_i \cdot n_i]}$$

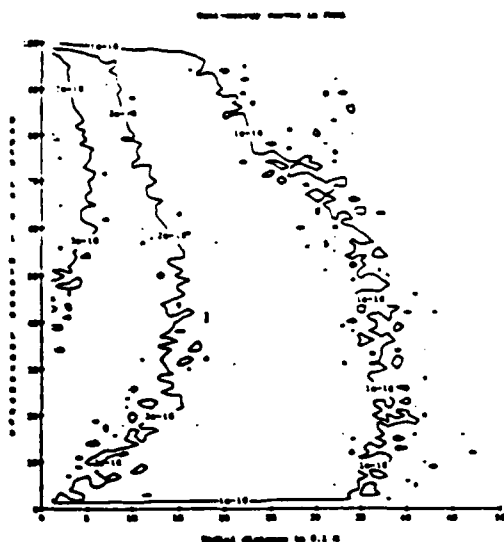
for an atom of species i , scattering cross section σ ,
and number of atoms per unit volume n .



Monte-Carlo simulation layout



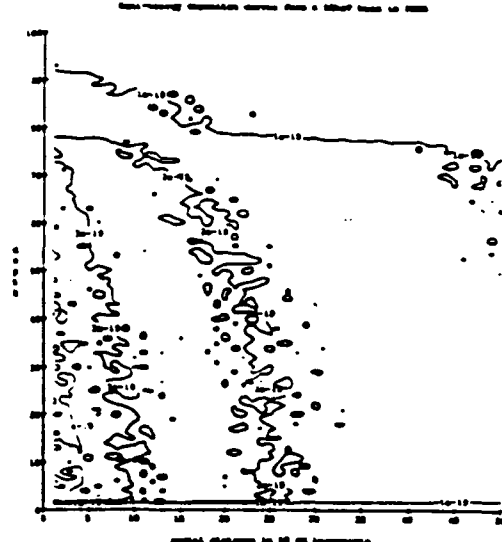
Setup for the Monte Carlo simulation.



Legend for Energy Deposition Curves for a 10μ Block of PMMA:

- 100 keV
- 50 keV
- 25 keV
- 10 keV
- 5 keV
- 2 keV
- 1 keV
- 0.5 keV
- 0.2 keV
- 0.1 keV

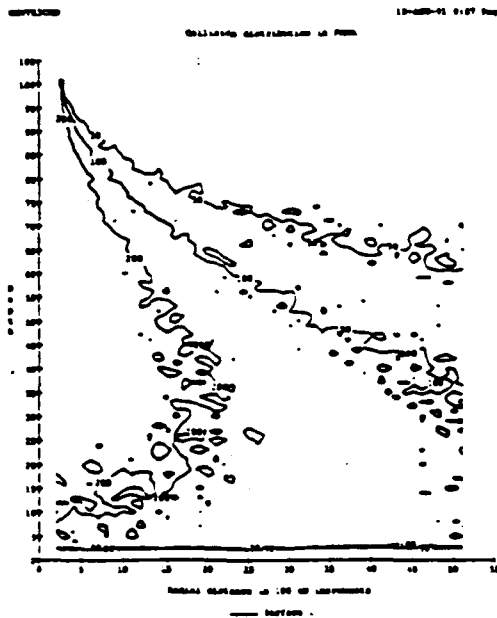
Energy Deposition Curves
for a 10μ Block of PMMA



Legend for Energy Deposition Curves for a 1 μ Block of PMMA:

- 100 keV
- 50 keV
- 25 keV
- 10 keV
- 5 keV
- 2 keV
- 1 keV
- 0.5 keV
- 0.2 keV
- 0.1 keV

Energy Deposition Curves
for a 1 μ Block of PMMA



Contour of electron-atom interactions
in a 10μ block of PMMA

Effect of a Sub-Micron Grating on Coherent Light

The theoretical effect of a sub-micron grating upon the coherent, monochromatic output of a 680nm laser was computed using the classical laws of optics. The purpose of the test was to allow a method of determining whether a given diffraction grating had been manufactured to specifications. The use of a laser was preferable to other methods of testing such as scanning electron microscopy due to its cheapness, reliability, ease of use, and accuracy. The use of the diffractive interference pattern produced when laser light is shone through the diffraction grating allows testing of the grating below the limit of resolution of optical microscopes.

The theory behind the test is based on the interaction of multiple but identical waves of light. To determine the behavior of the grating, a simulated beam of coherent, monochromatic light is shone through the grating and projected upon a screen parallel with the plane of the grating. The grating is composed of parallel metallized lines and clear spaces of equal width. The amplitude of the light at a point on the screen will then be the sum of the amplitudes from the wavefronts projected from each space on the grating,

$$A_{\text{sum}} := A_1 + A_2 + A_3 + \dots + A_n \quad (1)$$

The A_n 's are the amplitudes of the sinusoidal wave

$$A_n := A_0 \cdot \sin(\omega t + n \cdot \phi) \quad (2)$$

and the effect of these interfering wavefronts is given, when the A_n 's form an arithmetic progression, by Ditchman [7] as

$$A_{int} := A_0 \cdot \frac{\sin \left[n \cdot \pi \cdot \varepsilon \cdot \frac{\sin(\beta)}{\lambda} \right]}{\sin \left[\pi \cdot \varepsilon \cdot \frac{\sin(\beta)}{\lambda} \right]} \quad (3)$$

where ε is 1/2 the line-line separation, λ the wavelength, and β the angle of separation from the center of the grating,

$$\beta_n := \frac{x - n \cdot L}{\sqrt{s^2 + (x - n \cdot L)^2}} \quad (4)$$

taken at line number n where L is the total line+space distance and x is the distance along the plane of projection.

The wavefront amplitude A_0 coming from each of the slits is, however, modified by the diffraction of the incident wavefront off each of the line-space edges. The one-slit diffraction [8] is given by the relation

$$A_{diff} := A_0 \cdot \frac{\sin \left[2 \cdot \pi \cdot \delta \cdot \frac{\sin(\beta)}{\lambda} \right]}{2 \cdot \pi \cdot \delta \cdot \frac{\sin(\beta)}{\lambda}} \quad (5)$$

where A_0 is the initial amplitude of the beam and δ is half the width of the space.

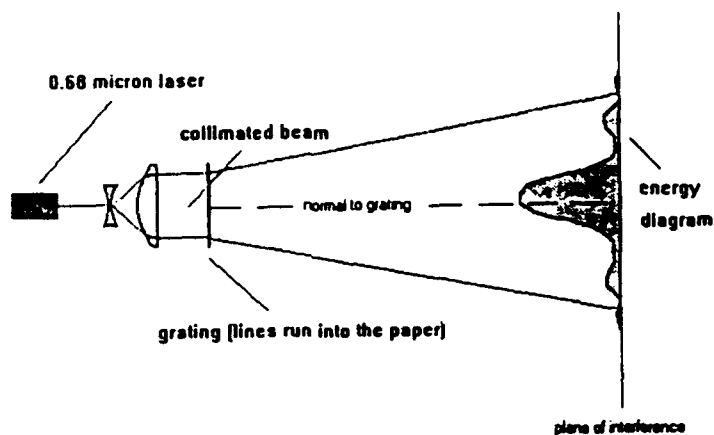
The total amplitude may then be found by substituting eq. (5) into A0 of eq. (3) and eq. (4) into the value β of eq. (3) for the final relation,

$$A := A_0 \cdot \frac{\sin \left[2 \cdot \pi \cdot \delta \cdot \frac{x - n \cdot L}{\lambda \sqrt{s^2 + (x - n \cdot L)^2}} \right]}{2 \cdot \pi \cdot \delta \cdot \frac{x - n \cdot L}{\lambda \sqrt{s^2 + (x - n \cdot L)^2}}} \cdot \frac{\sin \left[2 \cdot \varepsilon \cdot \pi \cdot N \cdot \frac{x - n \cdot L}{\lambda \sqrt{s^2 + (x - n \cdot L)^2}} \right]}{\sin \left[2 \cdot \varepsilon \cdot \pi \cdot \frac{x - n \cdot L}{\lambda \sqrt{s^2 + (x - n \cdot L)^2}} \right]} \quad (6)$$

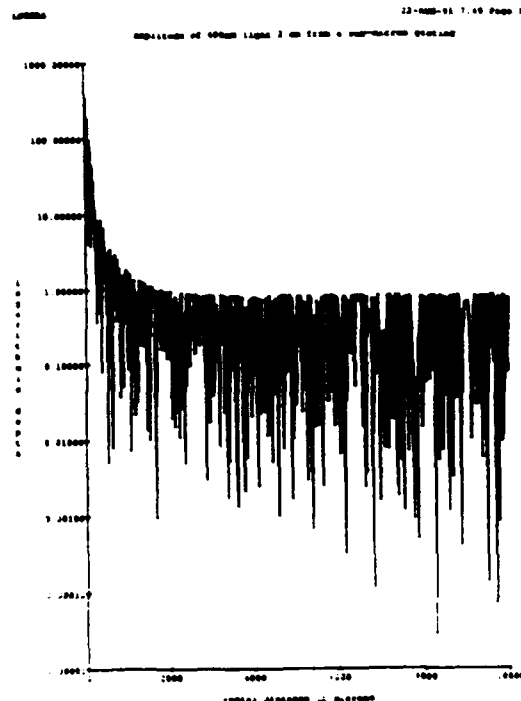
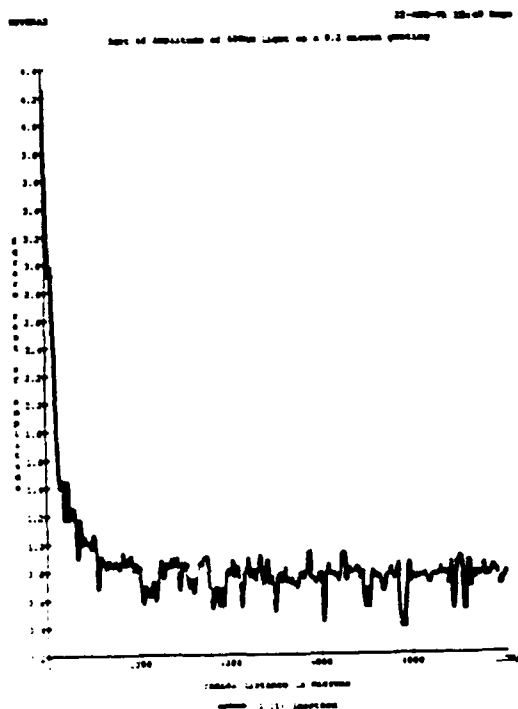
For the grating in question, the values are as follows:

N: 50000,
n: 25000,
 δ : 0.1 μM ,
 ε : 0.2 μM ,
L: 0.4 μM .

Eq. (6) was evaluated numerically. The results are shown on the following page.



Setup to perform a test--layout of the simulation.



Hazardous Chemical Labeling

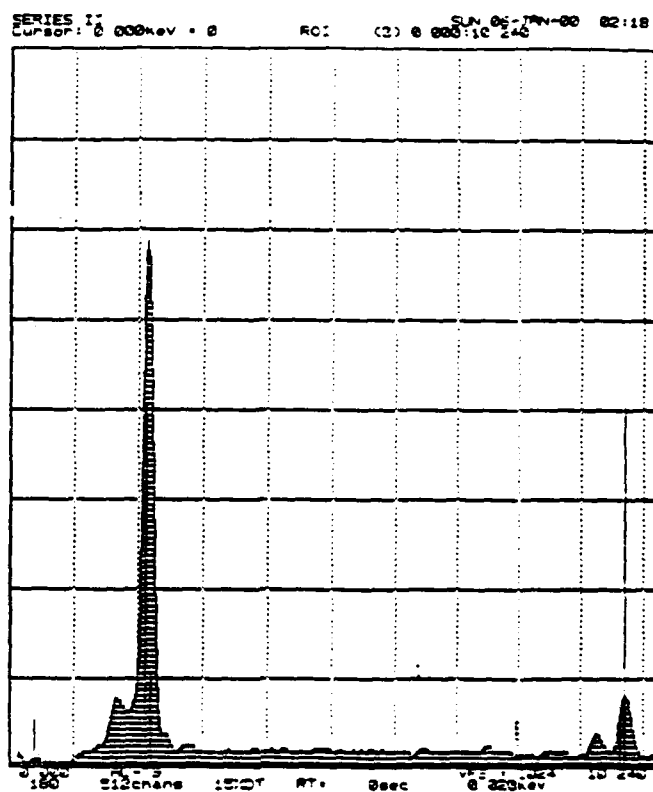
One problem with any credible research institute is that sometimes the researchers get caught up in major problems to the sacrifice of admittedly inconsequential chores. One assignment that merited my attention was the construction of hazard labels for laboratory chemicals. The labels were constructed under the Hazlabel program written by Matt Elwood of Cray Research. The program is a database which constructs labels for chemicals based on their C.A.S. numbers and allows identification of labeled bottles, emergency contact numbers, and dilution specifications. The labels are covered with a clear, acid resistant film which allows the use of grease-pencil updates. Each squeeze bottle and chemical-specific receptacle in the clean room facility received a label.

X-ray Backscattering Spectroscopy

The JOEL JSM-IC845A scanning electron microscope in the clean room facility is configured to run multi-element, semi-quantitative and qualitative analysis using X-ray detection through both light and heavy element windows. The X-rays, products of inelastic collisions between the incident electrons and the atomic species, are back scattered at energies corresponding to specific atoms. The Northern Tracer XBS was used to

evaluate both gold and bismuth samples.

Gold Spectra



Conclusions

The monte carlo simulations were carried out in PMMA in one and ten micron blocks and in 3-layer resists consisting of PMMA, MAA, and a mixture of the two, in one and ten micron blocks. The energy deposition profiles were obtained as given in the accompanying graphs. Additionally, the frequency of collision for electrons in PMMA is given as a separate contour. The results are close to those of other researchers in the field. The program designed to predict these contours was successfully completed and remains a working model. The next generation of program will include a menu or question driven environment to allow simplified calculations. The next step in the development of a useful model is the construction of a development model using the exposure values given in the monte-carlo simulation. A cellular model based on the designs of Samoto et al [6] is under construction. A combined exposure and development model will allow the prediction of the shape and size of profiles in resists as a function of the beam characteristics and resist variables.

The results of the sub-micron grating simulation have been prepared with no means of ascertaining their validity. The present system of electron beam lithography has proved incapable of producing the large and small scale nature of the grating to a degree of accuracy necessary for the test. The experimental

results of a real grating should be compared to those predicted when the grating finally becomes available.

Sources:

- [1]. B. Nigam, M. Sundaresan, and T. Wu, Phys. Rev. A 115, 491 (1959).
- [2]. R. Hawryluk, A. Hawryluk, and H. Smith, J. Appl. Phys. 45, 2551 (1974).
- [3]. K. Murata, T. Matsukawa, and R. Shimizu, Jpn. J. Appl. Phys. 10, 678 (1971).
- [4]. Bishop, H, Brit. J. Appl. Phys. 18, 703 (1967).
- [5]. Carter, G., and Grant, W, ION IMPLANTATION OF SEMICONDUCTORS. New York, Halsted Press, 1976.
- [6]. N. Samoto, R. Shimizu, and H. Hashimoto, J. Vac. Sci. Tech. B1 4, 1367 (1983).
- [7]. Ditchman, R.W., LIGHT. Interscience Publishers, Inc, New York, 1963, p. 181.
- [8]. Ibid, p. 175.

FINAL REPORT

Wright Laboratory
Solid State Electronics Directorate
Microelectronics Division

Helen Chou
Mentor: Luis Concha

20 August 1991

This summer I worked in the Microelectronics Division of the Solid State Electronics Directorate. My summer apprenticeship was aimed at learning computer operating systems. This was accomplished through several tasks chosen to compliment the ongoing work in the Microelectronics Division. I was tasked with becoming proficient with MS-DOS, the Unix operating system, and running a series of software benchmarking tests.

The Microelectronics Division is part of the Solid State Electronics Directorate at Wright Laboratory. The Microelectronics Division's mission is to direct and conduct contractual and in-house research in order to develop the technological base for advanced microelectronics. The division develops, demonstrates and applies a revolutionary foundation for state-of-the-art, high performance, reliable and cost effective Air Force weapon system electronics. Most of my time was spent in the Design Branch of the division. The Design Branch develops computer aided design (CAD) tools for the design and test of computer hardware.

As a summer hire for this division, I was tasked first to learn MS-DOS. MS-DOS stands for Micro Soft Disk Operating System. A disk operating system primarily helps a user create, keep track of, and manipulate files and directories. DOS provides more utility, including the ability to run applications programs such as word processing software. The method of training included review of the MS-DOS version 3.3 plus manuals combined with interactive use of an IBM compatible PC. After familiarizing myself by reading manuals and experimenting I was given a list of tasks to perform to demonstrate DOS proficiency. This experience was then utilized and augmented by

adding a word processing application program to my task list, WordPerfect 5.1. WordPerfect is a word processing software package used to create such documents as this paper. It was fairly easy to grasp and soon I found myself writing my biweekly logs using this software.

Task number two consisted of orienting myself with the Unix operating system. Since I had no previous experience working with computers, MS-DOS and WordPerfect were a good place to get experience before the jump to Unix. Unix is a complicated and powerful operating system. The method of training was the same as for DOS, independent study combined with experimenting and periodic requests - help! I was given four different user guides for the SUN Workstation. Each of these manuals was a step-by-step instructor that taught me all the commands needed to run programs on the SUN. In addition to the new commands and different file system, Unix on SUN computers provided a windows environment. This allowed several windows to be "open" simultaneously. This provides the user a multi-tasking capability. I was able to have several sessions going at one time, each doing a separate function. Finally I began writing shell scripts. A shell script is a list of commands to be performed in sequence. This is a very convenient way to execute repetitious command sequences. They are customized for specific needs including accepting different arguments for each execution. Scripts quietly run in the background allowing the user freedom to perform other tasks while waiting for completion.

Task three was the culmination of my summer internship, running a series of VHSIC Hardware Description Language (VHDL) software benchmarking tests. Benchmarking is a way to test software or

hardware with a set of "known" examples. They could test a computer language to demonstrate that it was complete and reacted normally to all features of the language. They could test a new implementation of a computer language on a specific computer. The benchmarks I ran were a combination of the software and hardware types described above.

The benchmark tests that I ran, tested commercial VHDL toolset features necessary so that government bases such as WPAFB are able to choose which computers and programs would be more valuable to purchase for the laboratories. The benchmark tests' results that were run on the SUN computer will be used in comparison to new computers going into the lab to see which ones are more efficient in doing certain jobs. The comparison system for this set of benchmarking had two variables - the computer system the tests were to be performed on and the vendor marketing a VHDL implementation for that computer. A typical benchmarking scenario goes like this: Vendor A and Vendor B each have two versions of VHDL for sale - one for the SUN computer and one for the VAX computer. We take a complete set of benchmarks and run that same set on each of the four possible combinations - Vendor A VHDL on the SUN, Vendor A VHDL on the VAX, Vendor B VHDL on the SUN and finally Vendor B VHDL on the VAX. The resulting data shows how each vendors product preformed on each machine: How much cpu time was required to execute the same tests? Were there any problems or peculiarities with specific implementations? In this way the Design Branch is able to advise other Air Force agencies what is best for their needs considering cost, size of the job to be performed, and current computer resources.

The last few weeks, I concentrated on running approximately 330 benchmark tests on the SUN Workstation. The benchmarks I used were

from the VHDL Benchmark Suite. They were a set of VHDL models each designed to test one or more specified features of the language in terms of system architecture, operating system, VHDL toolset model generating, building, and simulating a model. The tests each had one or more parameters that could be varied to observe their effect on timing and memory usage. I was able to run the benchmark tests with different parameters. Each test required the user to input a series of command lines. I learned a lot from this experience through hands-on-experience. I was able to understand the capacity of the computer. Each test had to be run separately or the whole system would malfunction. The system was very sensitive even to the slightest outside interferences. Each test, depending on the parameters used (large parameters required a longer time to run) took about 5-15 minutes each. I ended up spending approximately two and a half weeks running and rerunning each of the 330 benchmark tests with different parameters. Once my task was done, I had to go through and run each test through a timing file designed by Captain Karen Serafino from ELE department. Each of the tests that I had previously run had at least a page and a half of output information. The data of interest from the text are the timing lines dealing with the CPU times (Central Processing Unit times). The timing file that I used scanned the output files and collected the important data and wrote it to another file. This timing file contained the time it took the computer to execute each test.

This work was a totally new experience for me. I had never worked in an office before. I was very glad to see that the office environment was very relaxed and comfortable. During my eight weeks at WPAFB, I learned a lot about computers and working with people in a

professional atmosphere. I gained a degree of familiarity with MS-DOS and UNIX. I mastered WordPerfect which will be helpful throughout my schooling for writing research papers and other reports. I ran benchmarks to give the government needed information on other companies' computer products better enabling them to choose the better purchase. I learned how very important it is to have a good working rapport with co-workers and management. The working world is very different from the school that I am accustomed to. You are expected to do your part as a team member. Everyone on your team is affected by individual productivity.

I would like to take this opportunity to thank the following people: RDL and WPAFB for offering me such a great opportunity, not only to get valuable and memorable experience in a professional working atmosphere, but also giving me a chance to learn how important it is to get along with co-workers and management and to make a lot of great friends. I would like to extend my many thanks to my patient and understanding mentor, Luis Concha, for giving me guidance and the necessary tools in order for me to accomplish my projects this summer. I would also like to thank John Borger for offering his programming expertise; Dave Cartmell for his help with the UNIX language; and Captain Mike Dukes for reviving my computer whenever it went on the fritz. Thank you again for this wonderful opportunity to work under the AFOSR Summer Research Experience program.

WHAT I DID FOR MY SUMMER VACATION

Mr. Alan M. Page

In this summer's apprenticeship program, R.D.L. conjectured that all the students would be involved in a typical research environment, with each student assigned a task and a date for completion. Instead, I learned some of the basics of electrical engineering; circuit theory, circuit design, and fabrication for both digital and analog circuits, as well as various technical skills such as bread boarding, soldering, component wiring, and circuit diagramming. I even received some background in optics by building a laser. I feel that this was a more fruitful direction to take. Not only has this apprenticeship given me a broad exposure to all facets of electrical engineering and thus a solid experience base to build career decisions upon, it has provided a good foundation for my educational future. This foundation is the product of Mr. James Grote (with whom I spent most of the summer working), and includes some engineering nomenclature applicable in all fields, an information base for better comprehension of discussions at college, and a working knowledge of the skills necessary to be an engineer. The other method of apprenticeship with one assigned task leads to only esoteric facts about one particular discipline. Because I was not assigned a major task but many small ones, this paper will not be in the standard research format, but instead shall be presented as a synopsis of my achievements on a day to day basis.

I started off rather quickly as an apprentice engineer in the Electro-Optics Division at Wright-Patterson Air Force Base. By the end of the second day, I had already learned the use of an oscilloscope, signal generator, and pulse generator as well as finishing a laser safety computer course. I also experienced part of engineering life in the form of a briefing from the Electronics Directorate on computer privileges and the mission of Wright Labs. Two days later, I had built a active low-pass filter based on an operational amplifier from given equations, ran a signal into the filter and recorded output data, and experienced the medical portion of a laser lifestyle with an eye examination. As I moved from equation to finished product with the low-pass filter, Mr. Grote informed me that this is often the job of an engineer -- to take known formulas and parts, and then adapt them to the changing needs of the project at hand. On a simple level, this could correspond to a filter with a differing cut-off point, while on a more complex level could mean an optical

detector which transfers a signal to point A upon receiving wavelength A, or point B for wavelength B. Given formulas, an engineer would then design the detector for specified wavelength values. Finally, I plotted the output data from the filter using both log paper and a computer program called Math CAD, and also learned valuable information about the job of an engineer.

The second week, I constructed a summing amplifier (also utilizing an operational amplifier) and used it to send multiple signals through the low-pass filter, viewing the output on the scope. I designed and built a high-pass filter in the same manner as the low-pass, and also plotted the output data of the high-pass filter on the computer. I learned to solder with one of the technicians in the machine shop, and then began a major project -- a regulated 5 volt, 1.5 amp power supply. After reading extensive background information, I picked the necessary components, including one very large capacitor. This provided another lesson; oftentimes, if a component cannot be located, the cannibalization of outdated equipment is allowable. Soon I was drawing both circuit and wiring diagrams for the power supply, and then drilling, wiring, and soldering the circuit board. I drilled the box to contain the board and a transformer, and used a nibbler to cut square holes. Next, I soldered the exterior components, labeled the box with stick on letters, and put everything together (even the rubber feet!). After testing the output voltage at various currents (5.04 volts with .2 volts of ripple at 1 ampere), I finished my paperwork and began clean-up. I had learned all the steps a device goes through, from original engineering design through technical assembly, as well as valuable skills like soldering, practice with metal drilling and nibbling. In addition, I can now be more sympathetic towards the job of a technician.

My next project involved a Helium-Neon Laser (LASER is an acronym for Light Amplification by the Stimulated Emission of Radiation). Given a plasma tube, various mirrors with curvatures of 6 meters, 14 meters, or infinite curvature (flat), and a power supply, along with various equations for stability, I then endeavored to get the tube to lase. This involves very tedious, exacting positioning of the mirrors in mounts, along with mathematical proof that the laser is stable with the selected mirror curvatures. I succeeded in lasing every time except for the combination of two infinitely curved mirrors. From this experiment, I learned about the design and construction of lasers, laser stability, mirror curvature, and the problems inherent in tube gain. Additionally, I was given the feeling of being an actual electro-optic engineer. Finally, everything was reinforced by a textbook on lasers.

Lasers weren't the only textbook work; I also used them for digital circuits. AND, OR, NAND, and NOR gates make up the digital family, and I worked equations, picked components, and built circuits within this family next. My first digital construction was a combination of the simple gates built to resemble a decoder, but for one line segment only. I ran a signal through it, and detected a problem. My dismay at this can well be imagined, for up until this point everything had worked correctly the first time I tried it. This was just another object lesson that not everything works as desired, but without these little flaws, the engineering discipline probably wouldn't exist. After rebuilding the problematic circuit, I noticed an undue amount of distortion and used Schmitt triggers to relieve the same. Next, I designed a circuit capable of measuring the delay time of a hex inverter, but the scope was not fast enough to read the output. Also at this time, I learnt another advantage of engineering as we successfully used bandpass filters to safely view the solar eclipse.

The summer students received a tour of the directorate's world-class level clean room, which contains both class 100 and class 10 rooms. In a nylon suit, boots, hood, face mask, and bearing rubber gloves, we proudly ventured forth to view scanning electron microscopes (S.E.M.), molecular beam epitaxial growth machines (M.B.E.), and all the steps involved in making a wafer. All that cutting edge technology was both interesting and appealing.

Next, I used a 555 timer to build an oscillator based upon transistor-transistor-logic (T.T.L.). After the proper components were located, I had the capacitors and resistors necessary for a one hertz, 50 percent duty cycle output signal. I wired a counter, a 3-to-8 line decoder and a seven-segment LED display next to the 555 timer, and I had built a digital clock! I now know how clocks, watches, and auto-fire joysticks work. In addition, I feel more confident of my ability to design things I find interesting; like a super joystick or stereo components. That is what this program is about -- confidence and experience in a chosen field.

After a short stint with the technicians in the machine shop where I learned shop safety, the uses of the various tools such as the lathe, milling machine and drill press, as well as the traumas of not having blueprints, I ended the summer programming in machine language (hexadecimal) on a training system. This involved not only programming, but the building of circuits to interact directly with the computer. These were often very complicated, an example are memory circuits involving six I.C.'s. This also was a meaningful experience. I feel more confident about my

programming abilities in other languages and I absorb new information about a language more easily because I know why a given command works the way it does.

Eight weeks as an apprentice engineer has taught me much of the skills and interests necessary in the engineering field. I have learned circuit theory, design, and fabrication, soldering, bread boarding, and the use of generators and oscilloscopes, as well as various new and interesting computer programs. Beyond that, I have learnt many other things which are just as important. I have rediscovered the work ethic. I was told, "you can do anything or nothing any given day -- progress is measured only on a year to year basis." Because I like to see progress, though, I always strive to do something. When I was left alone on a project, I not only relearned drive and motivation, but I rekindled the desire for knowledge as well -- something I feel is very important for all college students, engineers, and everyone else in the world. I met and worked with many different people this summer, helping with on-the-job relations and social skills. By getting a broad exposure touching on all engineering topics and not having merely one major project, I feel I learnt more about the engineering discipline and its skills and interacted with more people than otherwise possible. Overall, however, I learned a lot, made a little bit of money, and had some fun on a truly novel job.

THE THIRD-ORDER INTERCEPT POINT IN MICROWAVE CIRCUITS

Suzette Yu, High School Apprentice

I. Summary

In order to improve efficiency and productivity, a program to pinpoint and plot the third-order intercept point was written. Among other tasks, this entailed computerized control of instruments like the power meter and spectrum analyzer during both single-input and two-tone tests that resulted in graphs demonstrating power output as a function of applied power. From this could be calculated the intercept point, which aids in defining the nonlinear portion of test amplifiers and its effect in system performance.

II. Introduction

Though such electronic equipment as wires and resistors are a common sight for many, other types of equipment geared toward different signals play a lesser-known but still important role in the realm of advanced technology. One such example of this involves microwave systems, which utilize frequencies (in the GHz range) that are higher than those standard in electronic circuitry. Implemented in amplifiers, signal controllers, sensors, and the like, microwave technology demands understanding of its principles and techniques if it is to be utilized to effectively advance development of such needed applications.

Similarly, before addressing specific issues dealing with high-frequency components, a general overview of instruments is needed. Relevant to the matter at hand are the oscillator, attenuator, directional coupler, power meter, and spectrum analyzer. The first of these, the oscillator, generates microwaves through various methods, the most common of which is that used in the Gunn oscillator. Here a dc voltage applied to a gallium arsenide crystal yields an irregular electron field that promotes electron bunching. Regardless of method of generation, once the microwave signal has been produced, its power can be altered through the

use of the attenuator, a component that contains absorbing material and can be employed to regulate the power before the signal reaches the device under study. At some point in the set-up after the attenuator may be placed a directional coupler that couples a certain amount of the power to a secondary arm. Often this secondary line leads to a power meter which gathers readings through gauging the effect of the signal on a thermistor, or thermal resistor, the resistance of which is temperature-dependent. Also able to measure power, the spectrum analyzer offers a visual means by which to characterize the signal through a graph of power versus frequency. These latter two devices may be computer-controlled by means of an interface bus, which allows the transmission and reception of information. Used in conjunction with other devices, this equipment provides material with which to construct microwave circuitry that performs needed work.

III. Discussion of Problem

One such task is the characterization of amplifiers, integral in circuitry design and function. Due to the nature of their electrical components, these devices reveal both a linear and non-linear region on a plot of power output as a function of power input. Thus, while the amplifier may begin, in its linear region, by inducing a change in output power which corresponds with the input change, at a certain high input the output may fail to follow the established trend. At this point, during which the amplifier is progressing toward a state called gain compression (gain being the ratio of power out to power in), the power input must undergo a greater jump to produce the given output. This nonlinearity consideration proves essential in predicting amplifier function and hence in regulating eventual system performance.

Although gain compression may be employed to determine nonlinearity, as in the specifying of a 1-dB deviation from the linear region and the comparison of devices by utilizing this 1-dB compression point, another

widely used method involves measuring IMD, or intermodulation distortion, a product of the mixing of two different frequencies (f_1 and f_2) due to the nonlinear portions of the instrument used, in this case the amplifier. These IMDs reflect the influence of nonlinearity on the circuit. Of particular concern are the third-order intermodulation products, so named because the coefficients in their terms, $2f_1 \pm f_2$ and $2f_2 \pm f_1$, yield a sum of three. As revealed in Figure 1, two of these IMDs themselves present a problem because they are closer to the "fundamental" frequencies than any other intermodulation products, actually appearing within the bandwidth of the amplifier. Thus is distortion produced. In order to make rough predictions about third-order IMDs at varied power levels, a quantity called the third-order intercept point may be obtained. Defined as the intersection of the linear regions of P_{f_1} and $P_{2f_1-f_2}$ (assuming a state of full linearity), where P represents the graph on a plot of power out as a function of power in, this point supplies information that not only enables the minimization of third-order IMD, but also allows the characterization of the amplifier's nonlinearity, which, as mentioned above, is crucial to accurate measurement.

It thus becomes desirable to obtain a means by which to easily pinpoint the value of the third-order intercept point.

IV. Results

The initial step in the process requires the determination of the single-input (f_1) curve. As shown by Figure 2, the amount of power applied to the amplifier was found with the power meter connected to the arm of the directional coupler, which deflected a minute portion of power, approximately .01 of the actual amount, to be read. (However, due to the variability of this number, the testing required calibration of the instruments in order to determine the exact percentage of power coupled to the secondary arm so that power meter readings could be adjusted accordingly.) After passing through the amplifier under test, the power

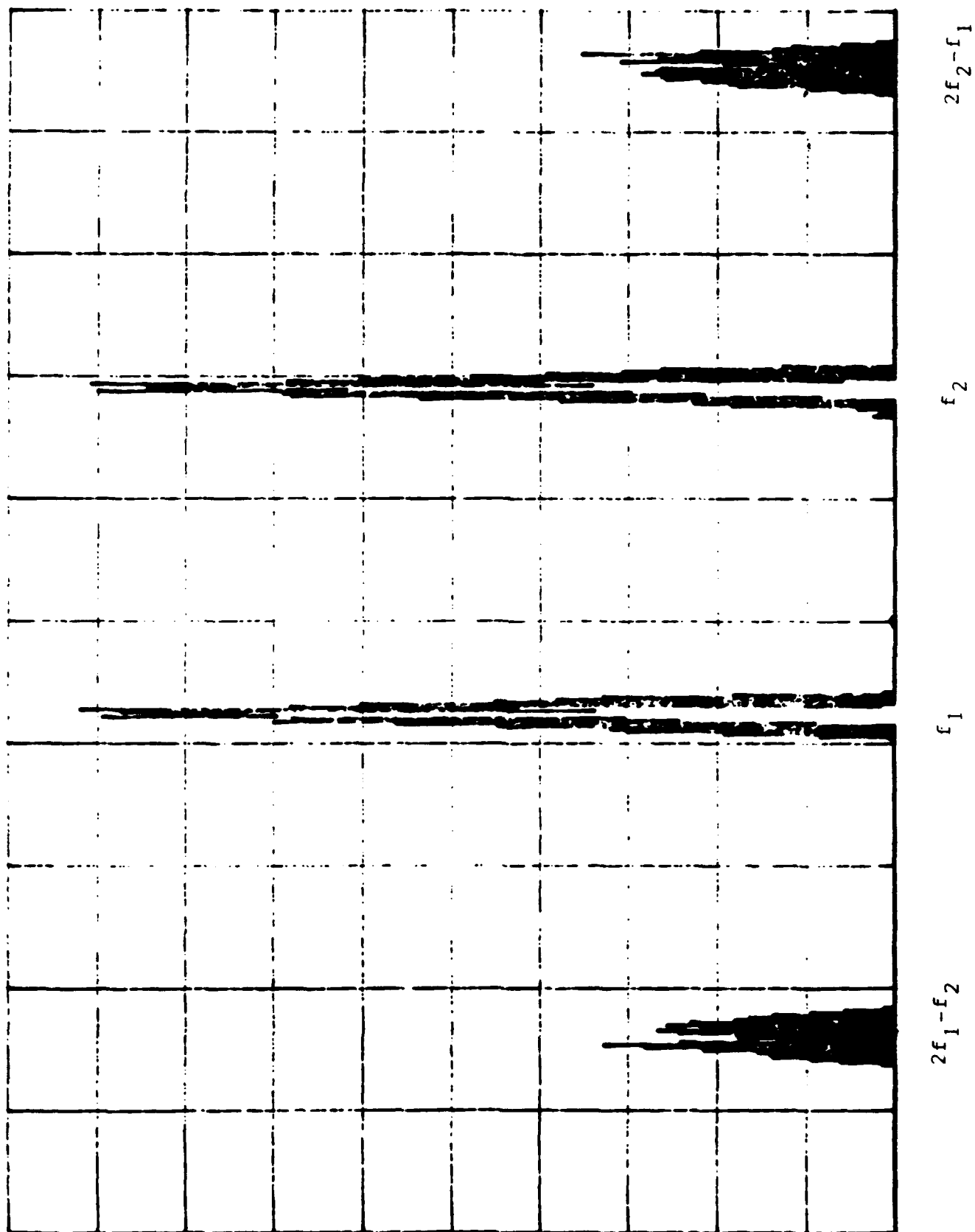


Figure 1: IMD on the spectrum analyzer

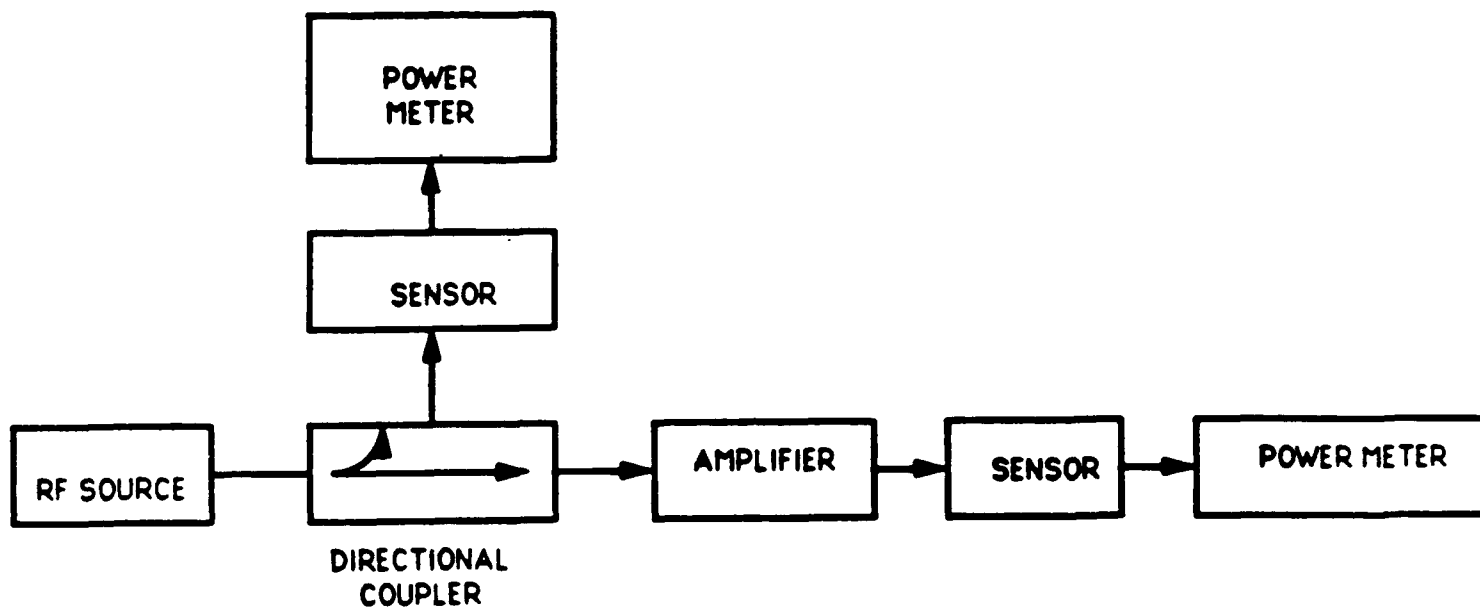


Figure 2: single-input test set-up

output was obtained with a second power meter. Through averaging of readings to yield data for plotted points and through incrementation of the attenuator, the information from these two instruments provided the graph of P_{in} versus P_{out} for the single-input part of the testing.

The two-tone, or dual frequency, test mandated a more complex circuit, illustrated in Figure 3. The methods, however, resembled those of the single-frequency runs, save that the spectrum analyzer replaced the power meter in finding output power. By placing on the $2f_1-f_2$ signal a marker that reported its value, the power out (ordinate) could be paired with power in (abscissa) to result in the curve for the IMD section of the graph.

Although data points could be manually recorded and plotted, a computerized option proves far superior in maximizing accuracy and efficient expenditure of time and resources. A program in HP BASIC, the system language that controlled the various devices on the test set-up, was written with the objective of determining the exact third-order intercept point as well as providing a graphical representation of the data. Through computerized manipulation of the power meter and spectrum analyzer, which yield data as described above, the program obtained the necessary information to plot power output as a function of input power for both single and two-tone frequencies. Because of the variable nature of the data, power-level dependent commands were required for establishing the necessary graphical environment. (Although the graphics features on other computer systems may dwarf those of the HP, the appearance of an immediate plot on the instrument-controlling system itself eliminates the hassle of transferring the data to another system and squandering time.)

Plotted amplifier curves notwithstanding, the definition of the third-order intercept point demands the extension of the linear sections of the curves until they intersect. But due to the imperfect nature of any data, even the so-called linear region may not represent an exact line from which an equation may be drawn, thus presenting a difficulty. This

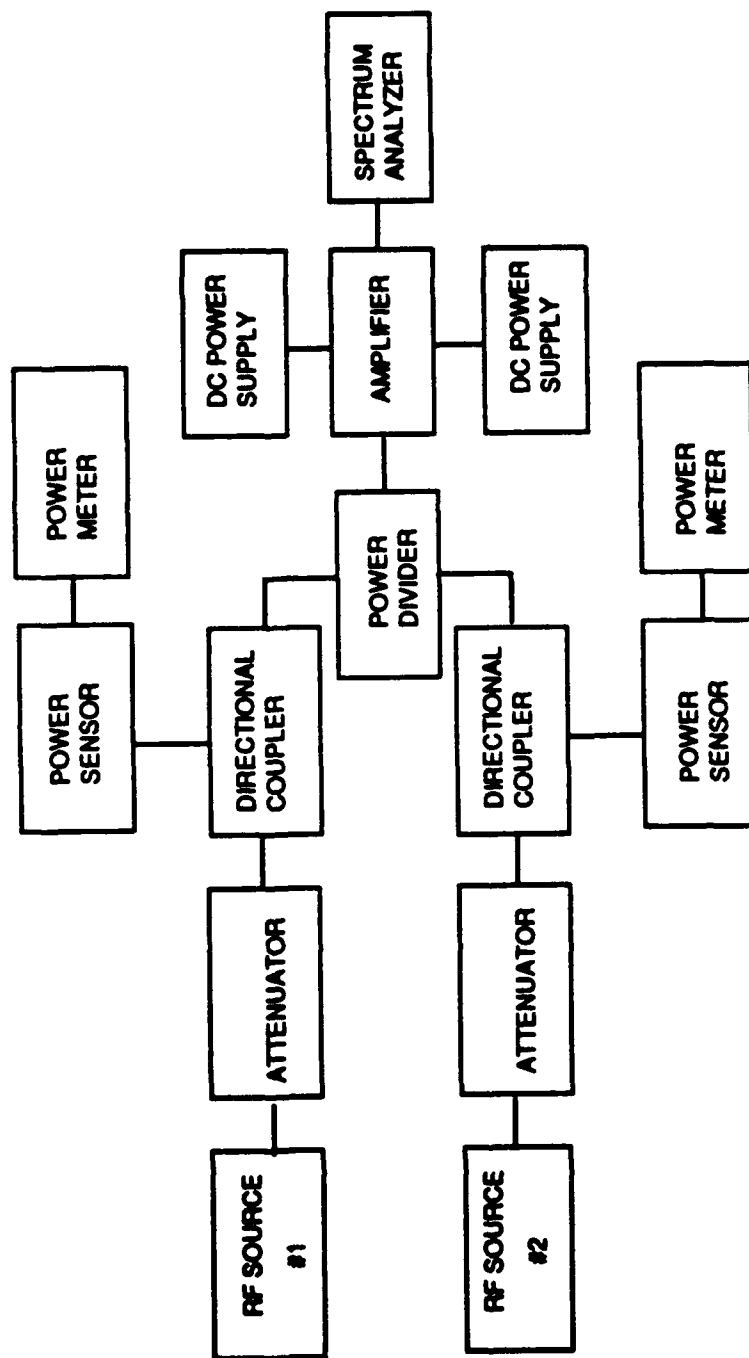


Figure 3: two-tone test set-up

situation was rectified through the use of linear regression, which coughs up a deterministic model given a probabalistic model based on actual data. Aiming to minimize the sum of squared errors (vertical distances between the deterministic line and data points), this least-squares approach produces a slope and y-intercept. Following the implementation of this procedure in the program, matrix operations were employed to generate the intersection of the two least-squares lines developed from the linear regions of the f_1 and $2f_1-f_2$ data curves.

V. Conclusion

The third-order intercept point pinpointed and plotted, it would seem that the original task has reached its final stages. However, as with all scientific endeavors, the apparent fulfillment of one objective simply establishes the basis for any number of subsequent insights. In this case, through the process of characterizing the linearity of the amplifier with the third-order intercept point, the program ensures a speedy method of defining the dynamic range, the values over which the amplifier acts in a linear manner. As mentioned above, such a function facilitates the acquiring of accurate results since it enables the avoidance of IMDs and other signals induced by the nonlinear region. And defining the third-order IMDs themselves proves helpful in anticipating distortion occurring when two signals are applied to an amplifier. Aside from these immediate benefits, the third-order intercept program represents a step in the possible eventual development of a model that would use such information as the intercept point to indicate the condition of other variables. Thus, while the program aids in the characterization of both amplifiers and third-order intermodulation distortion, it also opens the door to more complex endeavors which in turn may lay the basis for further scientific advancement.

Works Consulted

BASIC 4.0 Interfacing Techniques, Hewlett-Packard, Fort Collins, Colorado, 1985.

Cheung, W. Stephen, ed., and Levien, Frederic H., ed.,
Microwaves Made Simple: Principles and Applications, Artech House, Dedham, Mass., 1985.

Engelson, Morris, "Measuring IMD by Properly Using the Spectrum Analyzer,"
Microwaves & RF, Feb. 1988.

Gonzalez, Guillermo, Microwave Transistor Amplifiers, Prentice Hall, Englewood Cliffs, New Jersey, 1984.

Laverghetta, Thomas S., Practical Microwaves, Howard W. Sams, Indianapolis, 1984.

Scheaffer, Richard L., and McClave, James T., Statistics for Engineers, PWS Publishers, Boston, 1982.

HIGH SCHOOL APPRENTICESHIP FINAL REPORT

Jean Ay

All work that I did was independent, and most of my involvement was with a personal computer. I read through several manuals and began to learn how to program in simple BASIC. I experimented with spreadsheets and databases, and I used a draw program and an equation program to draw up a presentation. I also set up a UNISYS personal workstation and installed software for it. On occasion I checked the accuracy of computer printouts of equations.

CONTRACT WITH THE COMPUTER RESOURCES TEAM

Matt Becker

As an apprentice to Dick Smith in the Computer Resources Team, I have spent most of my first two weeks developing a comfortable working knowledge of the local computer environments. These skills will be used to transfer a computer graphics algorithm to a new, more useful system. The systems which I am using include Prime, Unix, and WordMarc. In order to familiarize myself with them I have read the various manuals and carried out a few small projects. As well as working on the graphics package some of my time has been spent setting up 13 new computers and installing their software.

The most important system which I will be using is called Prime. It's operating system (PRIMOS) is a distant brother of the Unix operating system. Among the many features of Primos, I have concentrated on the screen editor, command files, and Fortran language compilation. To begin the process of familiarization, my mentor asked me to analyze a Basic program that calculates free stream conditions encountered in a wind tunnel. Once I finished studying the source code, I began converting it to the language of Fortran. Using the screen editor to make the necessary changes made me aware of the many options and short-cuts available. In turn this knowledge will aid me in the much larger project of converting a whole graphics package. Also since the graphics project deals with large amounts of Fortran source code, this small conversion project gave me the chance to review Fortran as well as become familiar with the Fortran 77 compiler of Primos.

The other familiarization project consisted of learning to use command files. In general this process has been geared to using the command files for

long, tedious bind sessions which require the loading of each file into the linker. For the small calculation program such concerns were minor, but the large graphics package is a different matter. This is so because the files to be loaded number in the hundreds. Therefore the use of command procedure language (CPL) is essential to the fast conversion of the graphics program.

At this time, already 7 weeks into the project, I am continuing the difficult task of getting the graphics package up and running. My time has been spent in three stages. The first, described above, was the process of learning to use the basics of the computer software to which I had access. The second was learning to program in the language of C, because a number of subroutines in the package were in this language. The final and most difficult task was the actual conversion to Prime.

Included in this paper is the outline that I compiled as a guide to the language of C. It has come in very useful in these last few weeks, mainly because the Prime system does not have a C compiler and all the graphics subroutines written in C must be converted to Fortran 77. The outline includes the form of basic functions and structures in C, but since it is reproduced in full I will only say that I also wrote a test algorithm based on recursion and a bit manipulation program.

The final stage of my work here is the actual conversion. It began with the transfer of files from Silicon Graphic's Iris to Prime, a slow but essentially tedious task. Once they were moved the real excitement began. First each executable file was transferred into the same directory (BECKER>FORT.DIR>TRY1). Here I compiled the files and corrected errors as they appeared (common errors were the use of tabs). After everything compiled properly, I attempted to get an example program up and running. This process

was delayed by the fact that some of the user interface and bit operation subroutines were written in C, requiring conversion to Fortran 77. If space is available, I will include one of the five files that has undergone this revision. Another delay has come from the fact that the graphics package is not well structured, and the source code looks like spaghetti with the abundant use of goto statements and non-indentation format. This in turn has slowed the speed with which I can recognize the function of unfamiliar subroutines, and correct their errors. Even though the process is very difficult, it is not impossibly so. I have made some definite progress including the fact that one of the example programs will output a graphics metafile on the Prime system, which looks like it should when graphed on the SGI system. What remains to be done is getting the graphics translator running on Prime so the metafiles can be graphed without having to transfer them back to the SGI system. After that if I have enough time, I will begin another example program (which will go faster because the support programs are corrected) or I will work on getting the main program working.

These past 8 weeks have been very challenging in that I have been forced to acquire a great deal of knowledge in a short time in order to my job. I must thank my mentor, Dick Smith, for his continuous support and belief in my ability to do such a large task. Also, I wish to express my gratitude to Steve Scherr for being there when I had a question or needed guidance. I must say that the conversion of this graphics package, though not a research project, was definitely a learning experience which I shall not forget.

Short example of C
converted to
Fortran 77

```

/*****
*
*          Copyright (C) 1988
*      University Corporation for Atmospheric Research
*          All Rights Reserved
*
*          NCAR GRAPHICS V2.00
*
*****/

/*
    Logical operations for 32-bit systems.
*/

long
ishift(i, nshift)
/* shift the 4 byte integer i by nshift bits */
/* if nshift is negative, right shift end off zero fill */
/* if nshift is positive, left shift end around */
/* the routine behaves properly if magnitude of nshift > 32 */
    long      *i, *nshift;
{
    long      jshift, nbits;
    if (*nshift < 0) {
        nbits = (*nshift < -32 ? 32 : -*nshift);
        jshift = (*i >> nbits) & (01777777777 >> (nbits - 1));
    } else {
        nbits = *nshift % 32;
        jshift = (*i << nbits) | ((*i >> (32 - nbits))
                                   & (~ (03777777777 << nbits)));
    }
    return (jshift);
}

/* integer valued function to return logical AND of i and j */
/* i and j are assumed to be 4 byte integers */
long
iand_(i, j)
    long      *i, *j;
{
    return (*i & *j);
}

/* integer valued function to return logical OR of i and j */
/* i and j are assumed to be 4 byte integers */
long
ior_(i, j)
    long      *i, *j;
{
    return (*i | *j);
}

```

```

C*****C
C                                     C
C          Copyright (C) 1988         C
C    University Corporation for Atmospheric Research    C
C          All Rights Reserved        C
C                                     C
C          NCAR GRAPHICS V2.00        C
C                                     C
C*****C

```

```

C
C Set of three functions for logical operations on 32-bit systems
C

```

```

C
C Integer function ishift(i, nshift)
C - shift the 4 byte integer i by nshift bits
C - if nshift is negative, right shift end off zero fill
C - if nshift is positive, left rotate
C - the function behaves properly if magnitude of nshift > 32
C

```

```

    integer function ishift(i, nshift)
    integer i, nshift
    integer nbits
    if (nshift .lt. 0) then
        if (nshift .lt. -32) then
            nbits = 32
        else
            nbits = -nshift
        endif
        ishift = rs(i, nbits)
    else
        nbits = mod(nshift, 32)
        ishift = or(ls(i, nbits), rs(i, (32 - nbits)))
    endif
    return
end

```

```

C
C End of the integer function ishift
C

```

```

C
C Integer function iand(i, j)
C - integer valued function to return logical AND of i and j
C - i and j are assumed to be 4 byte integers
C

```

```

    integer function iand(i, j)
    integer i, j
    iand = and(i, j)
    return
end

```

```

C
C End of integer function iand
C

```

```
c
c Integer function ior(i, j)
c - integer valued function to return logical OR of i and j
c - i and j are assumed to be 4 byte integers
c
c     integer function ior(i, j)
c       integer i, j
c       ior = or(i, j)
c       return
c       end
c
c End of integer function ior
c
```

PROGRAMMING IN C

I. General Program Format

A. General format

```
1:  /* comments */
2:
3:  #include "stdio.h";
4:
5:  main()
6:  {
7:      executable statements
8:  }
```

B. Comments

1. comments begin with "/*" and end with "*/"
2. the markers don't have to be located on one line

C. Spacing, parentheses, and blank lines have no effect on program execution

D. Header files

1. the include statement for useful files

E. Functions

1. begin with a function name
2. followed by a parameter list in parentheses
3. the executable statements are contained in brackets
 - a. including input, output, and data manipulation statements
 - b. semicolons terminate statements

II. Data Types

A. Basic types

	:	range
1. (signed) char	:	-128 to 127
2. unsigned char	:	0 to 255
3. (signed) int	:	-32768 to 32767
4. unsigned int	:	0 to 65535
5. (signed) short int	:	-128 to 127
6. unsigned short int	:	0 to 255
7. (signed) long int	:	-2147483648 to 2147483649
8. unsigned long int	:	0 to 4294967296
9. float	:	3.4E-38 to 3.4E+38
10. double	:	1.7E-308 to 1.7E+308
11. long double	:	3.4E-4932 to 3.4E+4932
12. void	:	valueless

B. Access modifiers

1. modifier `data_type variable_list;`
2. `const`
 - a. receive value from initialization or hardware
 - b. cannot be modified in internal algorithmex. `"const float example = 40.20"`
3. `volatile`
 - a. volatile variable values checked each time referencedex. `"volatile int clock;"`
4. `volatile and const`
 - a. example can only be exchanged by external means

- ex. "const volatile char example = 'q';"
- C. Storage class specifier
1. specifier data_type variable_list;
 2. auto
 - a. specifies local variables
 - b. compiler assumes local inside a function making auto not necessary
 ex. "auto char sample, sample2;"
 3. extern
 - a. links global variables throughout multifile programs
 - b. within a file undeclared locals are checked for globality
 ex.

File one	File two
<div style="display: flex; justify-content: space-between; border-right: 1px solid black; padding-right: 10px;"> int x, y; extern int x, y; </div>	
 4. static
 - a. local static variables retain value after function end
 - b. local static are unknown outside of function
 - c. global static variables are known only inside file
 ex. "static int sum = 0;" (sum does not = 0 each call)
 5. register
 - a. stores variable in CPU, not memory (faster access)
 - b. at least two variables can be so declared
 - c. applies usually to local variables (parameters, too)
- D. Enumerations
1. enum enum_type_name {enumeration_list} variable_list;
ex. "enum apple {golden_del, red_del} fruit;"
 2. enumerations stand for integer values
 - a. values are incremental
 - b. if one is initialized, following ones are larger
 3. variable can equal only enumerators
 4. variables are incremented numerically through enumerators
 5. as with structures variables can be declared later
- E. typedef
1. typedef data_type name;
ex. "typedef float balance;
balance overdue;"
ex. "typedef struct client {element_list};
client clist;"
 2. allows new name to signify data type
- F. Type conversion
1. in expressions differing types are converted to larger type
 2. cast
 - a. (type) expression
ex. "int_num = (int)real_num;"
 - b. converts expression to given type

III. Variables

- A. declaration
1. data_type variable_name
- B. General info
1. first 31 characters recognized in variable name
 2. variables must be declared before they are used
 3. variables of same type
 - a. may be declared in same statement (seperated by commas)

- C. Local variables
 - 1. data_type variable_list;
ex. "char str1, str2;"
 - 2. declared within a function
 - 3. variables disappear at the end of their function {}
- D. Formal parameters
 - 1. call by value
 - a. declared as function parameters within parentheses
 - b. function_name (type var1, type var2, ...)
ex. "sub (int count)"
 - c. carry values into a function
 - d. variables are local to that function
 - 2. call by reference
 - a. pointer argument allowing change to variable everywhere
 - b. parameter declared as pointer type
 - c. must pass an address to pointer
ex. swap(&var)
 swap(int *point)
 - 3. call with arrays
 - a. array names without indeces are pointers to first element
 - b. parameter declaration (without indeces)
 - i) display(int num[10])
 - ii) display(int num[])
 - iii) display(int *num)
 - c. arrays /w indeces are treated as regular variables
- E. Command line arguments (parameters to main())
 - 1. main (int argc, char *argv[])
 - 2. argc : # of arguments on command line
 - 3. argv[1] : program name
 - 4. argv[2...] : subsequent strings
 - a. strings are seperated by tabs or spaces
 - b. double quotes surround one string
 - c. 2nd set of indeces refers to characters in the strings
 - 5. argc & argv can be substituted with any var_name
- E. Global variables
 - 1. data_type variable_list;
ex. "int sampl, samp2;"
 - 2. declared outside of all functions
 - 3. local variables with a global name
 - a. global value unaffected
 - b. local variable acts normally
- F. Variable initialization
 - 1. data_type variable_name = constant
ex. "float real_exmp = 1.89;"
 - 2. global variables default to zero
- G. Constants
 - 1. actual values like 'a' or '*' are constants
 - 2. assignment of constants to variables
 - a. assigned to type requiring the least memory
 - b. unless otherwise specified
 - i) long "L" after number
 - ii) float "F" after number
 - iii) unsigned "U" after number
 - 3. hexadecimal constants are preceded by "0x" (integers)
 - 4. octal constants are preceded by "0" (integers)

IV. Operators

A. Arithmetic ("arg1 operator arg2") (except c & d)

1. +, -, /, * : common normal operators
2. % : modulus (7%3 == 1)
3. ++ : increment (adds one at each pass)
 - a. y=x++ --> x=x+1 & y=x
 - b. y=++x --> x=x+1 & y=x+1
4. -- : decrement (same rules as increment)

B. Relational ("arg1 operator arg2")

1. > : greater than
2. >= : greater than or equal to
3. < : less than
4. <= : less than or equal to
5. == : equal to
6. != : not equal to

C. Logical

1. && : and
2. || : or
3. ! : not

D. Bitwise (arg1 operator arg2)

1. & : and (returns bit value of and statement)
 - a. turn off bits --> "ch = ch & 223;" (turns off 6th bit)
 - b. test bits --> "if(ch & 1)..." (checks 8th bit for on)
2. | : or (returns bit value of or statement)
 - a. turn on bits --> "ch = ch | 32;" (turns on 6th bit)
3. ^ : xor (returns on at bits that are different)
4. ~ : not (turns off on-bits, on off-bits)
5. >> : shift binary # right n bits (0 fill)
6. << : shift binary # left n bits (0 fill)

E. Miscellaneous

1. ? operator
 - a. var = exp1 ? exp2 : exp3;
 - b. var - variable to receive value
 - c. exp1 - comparison relation
 - d. exp2 - value of whole expression if exp1 = true
 - e. exp3 - value of whole expression if exp1 = false
2. comma operator
 - a. var = (exp1, exp2, exp3);
 - b. var - variable to receive value
 - c. exp1 & 2 - expressions to be completed from right to left
 - d. exp3 - value whole expression is to take
3. assignment operator
 - a. var1 = var2 = exp1
 - b. var1 & 2 - variables to receive value
 - c. exp1 - expression containing value
 - d. can be done with structures and unions of same type

E. Shorthand notation

1. var = var operator expression;
2. var operator= expression; (equivalent statements)

V. Print and Scan Statements ("stdio.h")

- A. printf("code_text", &variable_list); (& precedes variables)
- B. printf(array_name);
- C. scanf("control", &variable_list); (& precedes variables)

D. Backslash codes : meaning

1. \b : backspace
2. \f : form feed
3. \n : new line
4. \r : carriage return
5. \t : horizontal tab
6. \" : double quote
7. \' : single quote
8. \0 : null
9. \\ : backslash
10. \v : vertical tab
11. \a : alert
12. \N : N is octal constant
13. \xN : N is hexadecimal constant

E. Printf format codes (0 = wildcard)

1. %c : character
2. %d : signed decimal integer
3. %e : scientific notation of double (lowercase)
4. %E : scientific notation of double (uppercase)
5. %f : decimal floating point
6. %g : %e or %f (shortest form)
7. %G : %E or %f (shortest form)
8. %h0 : short data type 0
9. %i : signed decimal integer
10. %l0 : long data type 0
11. %L0 : long double (0 = e, f, g)
12. %n : returns to arg # of characters written
13. %o : unsigned octal of integer arg
14. %p : pointer
15. %s : string
16. %u : unsigned decimal integer
17. %x : unsigned hexadecimal of integer arg (lowercase)
18. %X : unsigned hexadecimal of integer arg (uppercase)
19. %% : percent sign
20. %000 : minimum field width specifier
 - a. if first 0 is 0 --> pad with "0"'s
 - b. if no first 0 --> pad with " "'s
 - c. second 0 = minimum field width (overruns exist)
 - d. third 0 = a regular format code
21. %0.00 : precision specifier
 - a. first 0 = minimum field width specifier
 - b. second .0 = maximum field width specifier
 - i) float point : # of decimal places
 - ii) strings : maximum length (no overruns)
 - iii) integer : minimum # of digits (overruns)
 - c. third 0 = a regular format specifier
22. %-00 : left justification in field of width 0
 - a. final 0 = a regular format code

F. Scanf format codes

1. similar codes: %c, %d, %i, %f, %o, %s, %x, %p, %n, %u
 - a. hexadecimal can be upper or lowercase
 - b. l & h & L modifiers work in same way
2. new codes:
 - a. %e : floating point
 - b. %g : floating point
 - c. %["0"] : reads 0 into arg & stops when char not same

- i) carot : looks for every thing but
 - ii) case sensitive
- 3. extraneous characters --> discarding those characters
- 4. white space --> discards all white space to next non
- 5. maximum field width = integer before letter of format code

VI. Library Functions

A. "stdio.h"

1. printf(stuff) : (see III)
2. scanf(stuff) : (see III)
3. gets(arg) : inputs string into given array
4. puts(arg) : displays string arrays & constants
 - a. recognizes format codes (no type conv)
5. getchar() : returns integer of keystroke buffer (echo)
 - a. problem with buffer which includes return
6. putchar(int) : displays argument as character
7. char *strstr(const char *str1, const char *str2)
 - a. returns pointer to first occurrence of str2 in str1
 - b. returns null if no match

B. "stdlib.h"

1. void abort(void)
 - a. terminates program (doesn't close files)
2. int abs(int num)
 - a. returns absolute value of num
3. rand() : generates a random integer (0-32767)
4. int atoi(const char *str)
 - a. returns integer converted from string
 - b. number terminated by non-digit character
 - c. returns 0 if no valid integer lies within str
5. double atof(const char *str)
 - a. returns double converted from str
 - b. number terminated by non-float character
 - c. returns 0 if no valid floating point lies within str
6. int atol(const char *str)
 - a. returns long converted from str
 - b. number terminated by non-integer character
 - c. returns 0 if no valid long integer lies within str
7. malloc() : (see XII)
8. free() : (see XII)
9. calloc() : (see XII)
10. realloc() : (see XII)
11. void exit(int status)
 - a. normal termination of program (0 is usual status)
12. long labs(long num)
 - a. returns absolute value of long int num

C. "math.h"

1. double sqrt(double arg)
 - a. returns square root of arg (arg>=0)
2. double acos(double arg)
 - a. returns arccos of arg (-1<=arg<=1)
3. double asin(double arg)
 - a. returns arcsin of arg (-1<=arg<=1)
4. double atan(double arg)
 - a. returns arctan of arg (-1<=arg<=1)
5. double atan2(double y, double x)

- a. returns arctan of y/x ($1 \leq y/x \leq 1$)
 - b. computes quadrant of return value
- 6. double ceil(double num)
 - a. returns smallest integer greater than num
- 7. double cos(double arg)
 - a. returns cos of arg (radians)
- 8. double cosh(double arg)
 - a. returns hyperbolic cos of arg (radians)
- 9. double exp(double arg)
 - a. returns e raised to the arg
- 10. double fabs(double arg)
 - a. returns absolute value of arg
- 11. double floor(double num)
 - a. returns greatest integer less than num
- 12. double log(double arg)
 - a. returns natural logarithm of arg
- 13. double log10(double arg)
 - a. returns base 10 logarithm of arg
- 14. double pow(double base, double exp)
 - a. returns base raised to the exp
 - b. if base = 0 then exp > 0
 - c. if base < 0 then exp = integer
- 15. double sin(double arg)
 - a. returns sin of arg (radians)
- 16. double sinh(double arg)
 - a. returns hyperbolic sin of arg (radians)
- 17. double tan(double arg)
 - a. returns tan of arg (radians)
- 18. double tanh(double arg)
 - a. returns hyperbolic tan of arg (radians)
- D. "conio.h"
 - 1. kbhit() : returns true if key was pressed, false otherwise
 - a. not standard on all systems
 - 2. getche() : returns integer of each keystroke (echo)
 - 3. getch() : returns integer of each keystroke
- E. "string.h"
 - 1. char *strcpy(char *str1, const char *str2)
 - a. copies str2 into str1
 - b. returns pointer to str1
 - 2. char *strcat(char *str1, const char *str2)
 - a. appends s2 to s1
 - b. returns pointer to str1
 - 3. int strcmp(const char *str1, const char *str2)
 - a. returns 0 if str1 == str2
 - b. returns positive integer if str1 > str2
 - c. returns negative integer if str1 < str2
 - 4. size_t strlen(const char *str1)
 - a. returns length of string (null not included)
 - 5. char *strchr(const char *str1, int ch)
 - a. returns pointer to first occurrence of ch in string str1
 - b. returns null if not found
 - 6. char *strtok(char *str1, const char *str2)
 - a. returns pointer to next token in str1
 - b. str2 make up delimiters separating tokens
- F. "ctype.h" (2-13 return 0 otherwise)
 - 1. int toupper(int ch)

- a. returns uppercase equivalent
2. `int tolower(int ch)`
 - a. returns lowercase equivalent
3. `int isalnum(int ch)`
 - a. returns non-zero if ch is alphanumeric
4. `int isalpha(int ch)`
 - a. returns non-zero if ch is a letter in the alphabet
5. `int iscntrl(int ch)`
 - a. returns non-zero if ch is a control character
 - b. $0 < ch < 0x1F$
6. `int isdigit(int ch)`
 - a. returns non-zero if ch is a digit
7. `int isgraph(int ch)`
 - a. returns non-zero if ch is printable character (no space)
 - b. $0x21 < ch < 0x7E$
8. `int islower(int ch)`
 - a. returns non-zero if ch is lowercase
9. `int isprint(int ch)`
 - a. returns non-zero if ch is printable character (spaces too)
 - b. $0x20 < ch < 0x7E$
10. `int ispunct(int ch)`
 - a. returns non-zero if ch is punctuation character
 - b. no spaces or alphanumeric characters
11. `int isspace(int ch)`
 - a. returns non-zero if ch is: space, tab, vertical tab, form feed, carriage return or newline character
12. `int isupper(int ch)`
 - a. returns non-zero if ch is uppercase
13. `int isxdigit(int ch)`
 - a. returns non-zero if ch is hexadecimal digit

VII. Selection, Iteration, and Jump Statements

A. If statement

1. `if(condition) { statement block }
else { statement block }`
ex. `"if (check==0) printf(" ");
else exit(0);"`
2. for condition to be true it must be non-zero
3. brackets required for two or more statements in block
4. nesting allowed for at least up to 15 levels
5. statement block omission requires ;

B. For loop

1. `for (initialization; condition; increment)
{ statement block }`
ex. `"for(cnt=0; cnt<max; cnt++) printf("\n");"`
2. loop executes until condition is false
3. two or more, init. or incr. statements separate with commas
4. each element of loop can be omitted (requires ;)
5. time delay loop
 - a. `for(x=0; x<=1000; ++x) ;`
 - b. null executables
6. infinite loops
 - a. `for(; ;) { statement block }`

C. Switch statement

1. `switch(variable) {`

- ```

 case constant1: statement sequence
 break;
 default: statement sequence
 }
 ex. "switch(test) {
 case 'y' :
 case 'Y' : printf(" ");
 break;
 default : exit(0);
 }"

```
2. break forces execution out of switch
    - a. code even in other cases is executed until break
  3. an empty case does not involve semicolons
  4. switch checks for equality - not comparison
- D. While loop
1. while(condition) { statement block }
    - ex. "while(true) {c=c/2; printf("%f", c);}
  2. statement block may be omitted, requires ";"
- E. Do-while loop
1. do { statement block }
    - while(condition);
    - ex. "do {key=getchar()} while(key!='\n');"
  2. forces one execution
- F. Continue & break
1. continue moves execution to condition test in loop
  2. break moves execution out of inner loop
    - a. if in a which, will effect which, not outer loop
- G. Loops may be nested at least 15 times

## VIII. Arrays and Strings

- A. Arrays
1. type var\_name[size][size2]...
  - ex. "int sample[10][3];"
  2. access from var\_name[0] to var\_name[size-1]
  3. no bounds checking can result in a crash if overwrite array
- B. Array initialization
1. type array\_name[size] = {value list}
    - ex. int sample[1][3] = {1,2,3};
  2. unsized array initialization
    - a. type array\_name[][size] = {value list}
    - b. C dimensions the array
- C. Strings
1. actually a character array
  2. initialization (two methods)
    - a. see array init.
    - b. type name[str\_size+1] = "string"
  3. value of null on the end of the string

## IX. Indirection or Pointers

- A. Type \*pointer\_name
- ex. int \*point
- B. Pointer operators
1. & : returns the address of a constant or variable
    - a. sting constants are stored in string table and can be acces

- b. `point = "weeeh"` : `point` contains address of string
    - 2. `*` : returns the value at an address
  - C. Base type
    - 1. insures the proper number of bytes will be copied
    - 2. must be same type as what it is pointing to
  - D. Assigning values
    - 1. `*point = value`
    - 2. assigns value to the address of `point`
  - E. Operations involving pointers
    - 1. `point++` : increments pointer to the next base type
    - 2. `point--` : decrements pointer to the next base type
    - 3. `point+val` : increments pointer to the valth base type
    - 4. `point-val` : decrements pointer to the valth base type
    - 5. pointers can be used in comparison statements
  - F. Array manipulation
    - 1. `point = string` : assigns `point` to first position of string
    - 2. usually much faster calculations than indexing
  - G. Indexing pointers
    - 1. `point[n]` : points to nth data type
    - 2. does not have to be declared
  - H. Multiple indirection
    - 1. pointer to a pointer
    - 2. declared with extra astericks & same base type
  - I. Pointers to functions (see VII.F)
- X. Functions
  - A. Parameters (see II.C)
  - B. Return
    - 1. all functions except void may return values
    - 2. return value; (value = constant, variable, or expression)
    - 3. default is integer
  - C. Prototype (declaration before it appears)
    - 1. `type func_name ();` (include type function)
    - 2. type void prevents a return from going unnoticed
  - D. Include type specifier before function name
  - E. Recursion
    - 1. function calls itself
  - F. Pointers to functions
    - 1. `data_type (*pointer)(parameter_list);`
    - 2. contain memory address of function entry
    - 3. address of function accessed by removing parentheses  
ex. `(*pointer) = example_function;`
    - 4. used to create generic functions
- XI. File I/O
  - A. Declaring a file pointer
    - 1. `FILE *pointer_name;` (file capitilized)
  - B. Functions "stdio.h"
    - 1. `fopen("file", "mode");` returns pointer to file (null on error)
    - 2. `fclose(pnt);` returns 0 on successful close
    - 3. `putc(ch, pnt);` writes charcter ch at pointer
    - 4. `getc(pnt);` returns character at pointer
    - 5. `fputc = putc & fgetc = getc`
    - 6. `feof(pnt);` returns true at end of file

7. `fputs(str, pnt);` writes `str` at pointer
8. `fgets(str, len, pnt);` reads into `str` at pointer until `len - 1` or `eoln` (`/n` is in string)
9. `rewind(pnt);` resets pointer to file start
10. `ferror(pnt);` returns true if error in last file oper.
11. `remove(file);` returns 0 on successful delete of file
12. `fflush(pnt);` returns 0 on successful flush of stream
13. `fread(pntl, bytes, items, pnt);`
  - a. `pntl` - address into which info is read
  - b. `bytes` - # of bytes per item
  - c. `items` - # of items to be read
  - d. `pnt` - pointer to file
14. `fwrite(pntl, bytes, items, pnt);` (see 13. read-->write)
15. `sizeof(var_type);` returns # of bytes for var or type
16. `fseek(pnt, bytes, origin);` returns 0 on succesful seek
  - a. `pnt` - pointer to file
  - b. `bytes` - # number of bytes after origin
  - c. `origin` (macro):
    - i) `SEEK_SET` : file start
    - ii) `SEEK_CUR` : current position
    - iii) `SEEK_END` : end of file
17. `fprintf(pnt, "control", args);` (=printf only w/ pointer)
18. `fscanf(pnt, "control", args);` (=scanf only w/ pointer)
- C. Modes (\* = add to 1, 2 and 3)
  1. `r` : open (read)
  2. `w` : create (write) [destroy previous file]
  3. `a` : append to
  4. `*` : a text file
  5. `*b` : a binary file
  6. `++` : for read and write
  7. `*b+` : a binary file for read and write

## XII. Structures & Unions

### A. Purpose

1. structures unify similar data into coherent unit
2. union allow easy type conversion

### B. Stucture Declaration

1. `struct type_name {`  
     `type element_name1;`  
     `...`  
     `type element_nameN;`  
   `} struct_var_name1, ..., struct_var_nameN;`  
   ex. `"struct ex_type {`  
       `char ch[80];`  
       `int bong;`  
       `} var1, var2, arr[15];"`
2. structure variables can be declared later
  - a. `struct type_name var_name;`  
   ex. `"struct ex_type another_var;"`
3. structure variables can be arrays
4. either `type_name` or `struct_var_name` can be omitted, but not bot

### C. Union declaration

1. same form as structure, except replace `"struct"` with `"union"`

### D. Referencing elements

1. `struct_or_union_var_name.element_name`



- ex. "ex\_type.ch[5] = 'p'"
  - 2. dot operator
- E. Structure parameters
  - 1. types must match (use global struct decl. or same parm decl.)
  - 2. call by value method
- F. Structure pointers (call-by-reference)
  - 1. declare like regular variable only with asterisk
    - ex. "struct ex\_type \*pointer;"
  - 2. when assigning an address, must use &
    - ex. "pointer = &ex\_type"
  - 3. arrow operator (->) accesses elements (compliment to dot)
- G. Nested structures
  - 1. use dot operator as needed to access nested elements
- H. Bit fields
  - 1. access to actual bits in a field
  - 2. type must be int, signed, or unsigned (single bit)
    - a. forces usage of at least two bytes for storage of field
  - 3. type element\_name : length; (within structures or unions)
    - ex. "unsigned bit\_hold : 1;"
  - 4. unnamed bits to ease access of later elements
  - 5. don't have to include later types not needed
  - 6. restrictions
    - a. do not have addresses
    - b. no arrays
    - c. can't overlap integer boundaries
- I. Unions
  - 1. large as largest element
  - 2. store data at same location in memory
  - 3. with bit fields provides method of decoding to binary

### XIII. Dynamic allocation

- A. memory usage
  - 1. permanent storage : program code & global variables
  - 2. stack : local variables
  - 3. heap : free space between #1 & #2
- B. void \*malloc(size\_t size)
  - 1. returns pointer to location of free memory
  - 2. returns 0 if memory not available
  - 3. pnt = (type \*)allocation()
    - ex. "pointer = malloc(sizeof(int))"
- C. void free(void \*ptr);
  - 1. opens memory for reuse
    - ex. "free(pointer);"
- D. void \*calloc(size\_t num, size\_t bytes)
  - 1. returns pointer to location of allocated memory
  - 2. returns null if not sufficient memory
- E. void \*realloc(void \*ptr, size\_t size)
  - 1. changes size of memory allocation at \*ptr to size
  - 2. returns pointer to new memory allocation at \*ptr
  - 3. returns null if insufficient memory

### XIV. Preprocessor Directives

- A. #define macro\_name(arg\_list) string
  - 1. substitution of string for macro\_name

2. no sub if macro\_name is within a string
  3. a space separates the macro\_name from the string
  4. string is terminated by a new line character
  5. macro arguments
    - a. replace occurrences in string
- ex. `#define MIN(a,b) "help" & (a>>b)`  
`if(MIN(mask, bits)) ...`  
`--> if("help"&(mask>>bits)) ...`
- B. `#undef macro_name`
    1. undefines macro\_name
  - C. `#error error message`
    1. stops compilation and prints error message
    2. don't need quotes on message
  - D. `#include "file_name" (or <file_name>)`
    1. includes file file\_name when compiling
    2. `"` force search in current dir and special dir
    3. `<>` force search in special dir
  - E. `#line number "file_name"`
    1. resets line counter to number
  - F. `#pragma name`
    1. loads instructions from compiler of name
  - G. `# string`
    1. makes following string a quoted string
  - H. `arg ## arg2`
    1. concatenates to arguments --> argarg2
  - I. Conditional directives
    1. `#if const exp1 statement sequence`
      - a. if exp1 = true then do sequence
    2. `#elif exp2 statement sequence 2`
      - a. else if exp2 true then do second sequence
    3. `#else statement sequence 3`
      - a. if previous exp = false then do third sequence
    4. `#endif`
      - a. concludes `#if`, `#ifdef` & `#ifndef` statements
    5. `#ifdef macro_name statement sequence`
      - a. if macro\_name is defined then it does statement sequence
      - b. uses only `#else` & `#endif` directives, not `#elif`
    6. `#ifndef macro_name statement sequence`
      - a. if macro\_name is undefined then do statement sequence
      - b. uses only `#else` & `#endif` directives, not `#elif`

## Saving Time, Money, and Lives with CFD

AFOSR apprentice Shari Goldenberg

Experimentation is an expensive, lengthy way of testing equipment. This is especially true in the Air Force where large amounts of equipment must be tested again and again in order to insure a crewman's safety when using them. Ejection seats are one of the many items that take an excessive amount of time and money to test. When I came to the Crew Escape Group of the Flight Dynamics Laboratory, I did not fully realize how much testing actually did occur before an ejection seat was produced. During my first week, I read a two volume book about the history of ejection seats which enabled me to understand these problems in more depth. First of all, money is spent designing and building an ejection seat. Next, one must test the ejection seat with a dummy seated inside which increases the cost further. Many times during these tests, the ejection seat does not work properly and is destroyed due to errors in the test procedure or in the design of the seat. This results in another large money expenditure for the express purpose of repairing and redesigning the ejection seat. One can see how this could become very expensive and time consuming after only a few tests fail. Another aspect one must look at when testing ejection seats is the possibility of human lives being endangered. In the past, when several of these dummy tests succeeded in a row, the dummy was taken out of the seat and a volunteer crewmember was used to test the seat more fully. Unfortunately, there were failures during this trial period as well, and many volunteer crewmembers were hurt, maimed, or killed. For example, on 22 February 1950, the F-89 ejection system was tested. During the test, the pilot realized that something was seriously wrong with the plane. He yelled "bail out!" to the

technician in the rear seat. The technician's ejection seat did not function properly, and he died in the resulting crash of the plane. Additionally, the pilot received a dislocated shoulder and a broken leg. His ejection seat had an operational error, but at least it propelled him out of the aircraft eventually unlike the technician's seat (reference a). Moreover, during another F-89 ejection system test on 28 June 1952, a crew chief died when he ejected from his aircraft due to a malfunction in the deployment of his parachute (reference a). These situations are a perfect example of how lives were lost or injured during testing. After several more unfortunate incidents like these where crewmembers were either hurt or killed, the engineers decided to incorporate a major redesign program for the F-89 ejection system. Again, this took a large amount of time and money to set up this program and provide the necessary equipment and personnel. When I finished reading about this ejection system and many more just like it, the lives lost, the time expended, and the expenses incurred during testing had a profound impact. The lives lost during testing had an effect on me, but it is not a valid concern anymore because different methods are now available for testing equipment without endangering people's lives. For example, one could use a wind tunnel to test an ejection seat. The Aircrew Escape Group, in addition to expanding the capability of existing escape systems, also endeavors to discover differing methods for analyzing and evaluating escape systems without using volunteers. Mr. Shawn Brown, a project engineer in the group, was and still is using CFD (computational fluid dynamics) to tackle this challenging aerodynamic issue. When I arrived, I was assigned to work with him on this project. Using CFD to analyze the flow field characteristics of an ejection seat/man would cut down on many of the tests done; therefore, less money and time would be spent in the

testing process. In my endeavor to learn the basics of CFD and apply it toward the design of future escape systems, I first had to study how an ejection seat operates as well as learn the fundamentals of the different computer systems.

Since, at the start, I had no background on ejection seats, my first goal was to read and learn the history behind the evolution of ejection seats. One of the many books I read was about the development of the Martin-Baker ejection seat (reference b). One of the first things Sir James Martin, co-founder of the Martin-Baker Co., discovered was the effect of g forces on a crewmember during ejection from an aircraft. He was the first to discern that an excessively high rate of increase of acceleration can cause serious spinal injuries. After many more studies were enacted, Sir James Martin found that the peak acceleration of the seat should not exceed 21g's, and it should not be endured for more than one-tenth of a second. Beyond that, he ascertained that the rate of rise of g should be no greater than 300g per second. He then discovered that when sustaining these g forces, the body should be held in a position that would ensure that the adjacent spinal vertebrae are square to each other. This reduces the probability that a crewmember will receive a spinal injury during the ejection sequence. In general, these factors are now considered to be the basic design criteria for ejection seats. Besides studying the effects of g forces, Sir James Martin also made many technical developments to the ejection seat. For instance, to eject itself out of the aircraft, the seat went from having a swinging arm to having a rocket catapult. This was done because a higher percentage of lives were saved by the use of a catapult as opposed to a swinging arm. In addition, Sir James Martin added a barostat onto the time release unit to prevent the seat from ejecting until the aircraft had

descended to an altitude where there was enough oxygen for the occupant in the seat to breathe unassisted. This book taught me the basics on the forces acting upon the seat/man combination as well as many of the technological advances made to the seat.

In addition to the book about Martin-Baker, I also read a technical report on the ACES II ejection seat (reference c). The primary reason I read this report was to achieve a better understanding of how an ejection seat operates and what components comprise it. The ACES II report discussed each part's function, and when this function occurred in the ejection sequence. The sequence starts off with the firing of the rocket catapult which shoots the seat/man out of the aircraft. Next, the drogue gun fires and pushes the drogue, a small parachute with a hole in the top, outward at a rapid speed. Then the drogue inflates to slow the seat/man down. The drogue is sent out before the main parachute so that the crewmember is falling at a slow enough rate that the shock of the main parachute opening will not injure them. After that, STAPAC, the pitch stabilization system, ignites to ensure that the ejection seat/man does not achieve an adverse attitude that would hinder the possibility of a safe ejection. Further on in the sequence, the seat and man separate when the mortar explodes which sends the main parachute out at a quicker rate than it would normally deploy. At this point, the reefer lines are cut, and the parachute inflates. Reefer lines hold the parachute closed until the drogue has sufficiently slowed the man down, so he will not suffer grievous injuries. Lastly, the survival equipment deploys, and the man makes his final descent to the ground. Another topic this report dealt with was the differing trajectories a seat/man combination might take. I learned that a zero-zero trajectory is a path one takes when ejecting from an aircraft at zero altitude and zero velocity.

When I finished this report, I was still a little unclear about how certain parts of the ejection seat worked; therefore, Mr. Giovanni Pagán, another group member, showed me some actual ejection seats and explained in further depth how they operated. By this time, I had a fairly good idea on what was involved in the ejection of an ejection seat from the moment of firing to the final descent.

With this background information on ejection seats and with help from Mr. Brown and Lt. Dave Flynt, I used CADKEY, a computer aided design software program, to digitize on a personal computer the outline of an F-106 ejection seat and man combination (see Diagram 1). We decided to do a two-dimensional drawing because that seemed the most practical and realistic at the time. We already had programs that would generate grids for the two-dimensional seat/man as well as a program that would tell us the fluid flow characteristics for a twodimensional object. Before I started to digitize the seat/man, I typed in an initial measurement of twenty-four inches for the back of the ejection seat. Therefore, the rest of the digitization was scaled by the computer to that first reference measurement. After the first digitized drawing which was as accurate as I could possibly make it, I made two more rounded, less detailed drawings (see Diagram 2). We did it this way in order for the grid to take on a better shape with better results. By digitizing this drawing, I learned a lot about using CADKEY and a personal computer.

After digitizing a two-dimensional seat/man, Mr. Brown determined that it was possible to do a three-dimensional seat/man drawing. The advantages to making a three-dimensional image were many. For instance, there is greater accuracy with a three-dimensional image as opposed to a two-dimensional image. Instead of receiving data as a general trend, one would be able to receive

extremely accurate information. For these reasons, Mr. Brown had searched for and found an old file with the nodes and elements of a three-dimensional figure. All we would have to do was type it into the computer using another software package called WordPerfect, a word processing program. Once we typed these nodes and elements into WordPerfect, we could then transfer that file into CADKEY in order to get a graphic display of what we had typed in. Mr. Brown had even found a program that would generate a grid around this three-dimensional object. Unfortunately, after all that work, we discovered that some of the nodes and elements in this old file were missing; therefore, we were rendered incapable of finishing this specific three-dimensional seat/man image. Although we did not complete this seat/man combination, I did learn how to use different parts of the WordPerfect software.

Once we had delved into the possibility of a three-dimensional seat/man, we could not give it up. Mr. Brown again came up with the idea of having me create a three-dimensional image from scratch. It would take a while, but it would be many times more accurate than a two-dimensional object. Again, we chose CADKEY as the software package that would be the most effective and useful for this type of three-dimensional object. First, we took measurements from the ACES II report to construct this seat/man display. For example, we took the measurement from the back of the ejection seat to the man's kneecaps as well as two to three other strategical places along the body. These became the outer boundary points of the seat/man. I then created lines and curves that would outline these boundary points to combine them into a general shape of a seat/man combination. We then put points onto these lines and curves for the express purpose of creating nodes from which a grid could be generated (see Diagram 3). After several weeks of working on this, it was finally finished. Next, we printed several different



views of the seat/man out on the laser printer. This printer is mainly used to print graphics and text. The only disadvantage is that it takes a long time to print out complex drawings. While working on this seat/man image, I learned about a laser printer and even more about CADKEY and its uses.

Once I had completed this drawing, Mr. Brown and I had to decide which grid generation method would be the best. From among the choices, we were seriously considering only two, either a hyperbolic or an algebraic grid method. For an algebraic grid, we would have had to create our own analytical equations such as using logarithmic functions to define the grid lines. Besides this taking a long time, the equation one eventually comes up with may not give a person the exact type of grid they wanted. Then, the person would have to take even more time trying to create another equation that will make the kind of grid they really want. To make things even harder, one has to add into the equation controls such as how much spacing one wants in the grid. Also, one would have to specify inner and outer boundaries for the grid. Although that does not sound difficult, it actually takes quite a while to determine what these boundaries are. For a hyperbolic grid, one uses hyperbolic partial differential equations to solve for the grid a person wants. Moreover, because this grid uses hyperbolic partial differential equations, the grid will automatically be formed normal to the surface of the ejection seat and man. Besides that, one only has to specify the inner boundary, and the approximate distance at which one wants the outer boundary of the grid. Even further, one can control spacing and other features of the grid by setting constants rather than having to add them into one's equation. Also, just recently developed was an efficient three-dimensional hyperbolic grid generator. With this program, one can put the whole figure of the seat and man in the program and receive accurate

results. Before this program, we would have had to generate two-dimensional grids of each cross section of the seat and man from top to bottom. Then, we would have piled these grids one on top of the other to come up with a three-dimensional grid. For the reasons mentioned above, we decided the most effective way to create the grid was hyperbolically.

Once we had decided on the type of grid method to use, we then had to generate this grid. Our first objective was to specify the parameters such as the artificial viscosity we wanted. We would have to experiment with these parameters until the grid was formed the way we wanted it. We would then take the output from the grid generating program and input it into the ARC-3D program. This program computes all the fluid flow characteristics that would act upon the ejection seat and man. One would have to input into the ARC-3D program many different factors such as the Mach number, the Reynold's number, the reference length, and more. Also, one would have to give ARC-3D the x, y, and z points of the grid which is basically giving the program the size of the grid. At this point, one would run the program and compare the computational output with the experimental results to see if what the program emitted was the same. In conclusion, using CFD will save time, money, and even lives if used effectively.

# Diagram #1

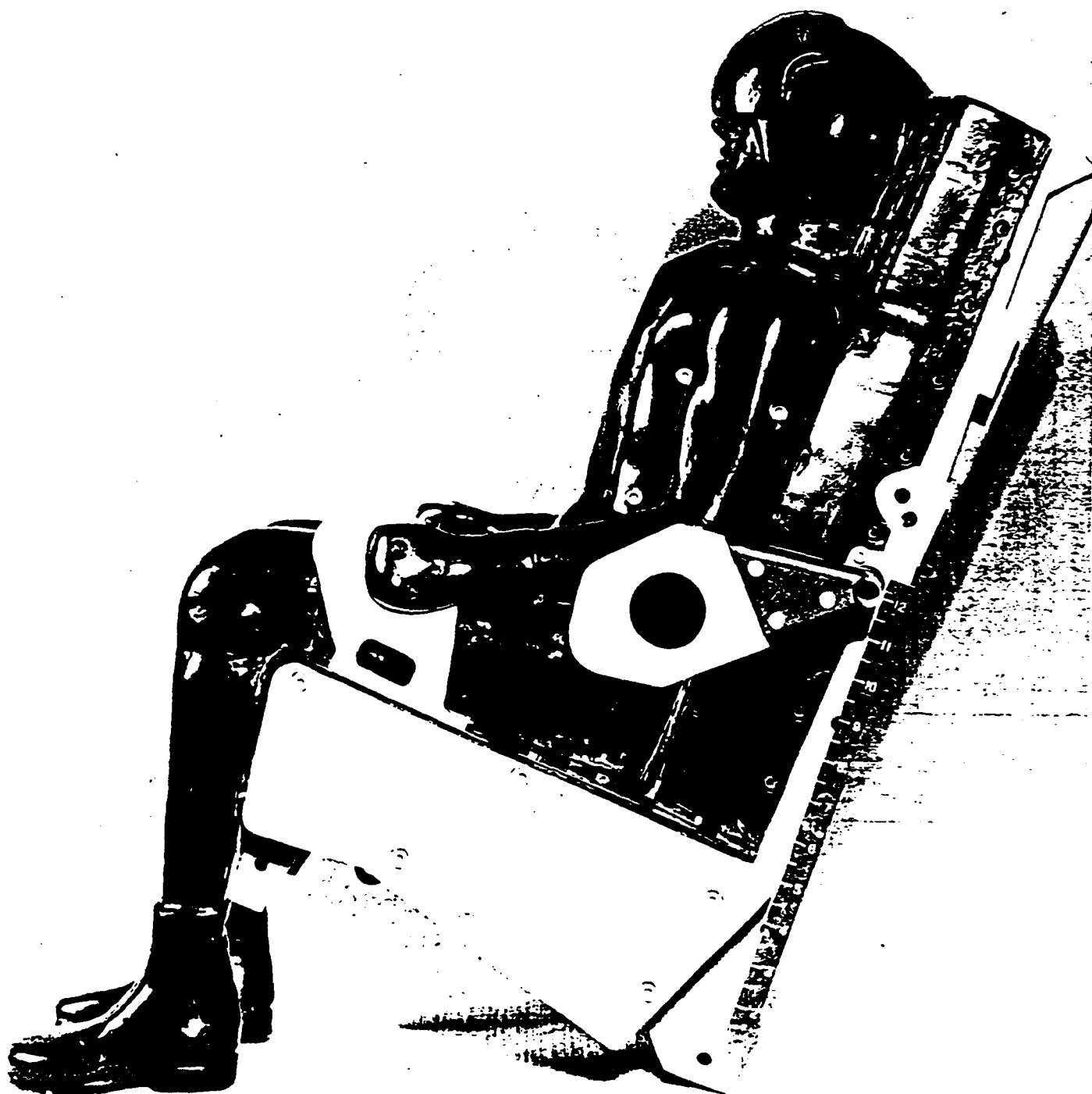
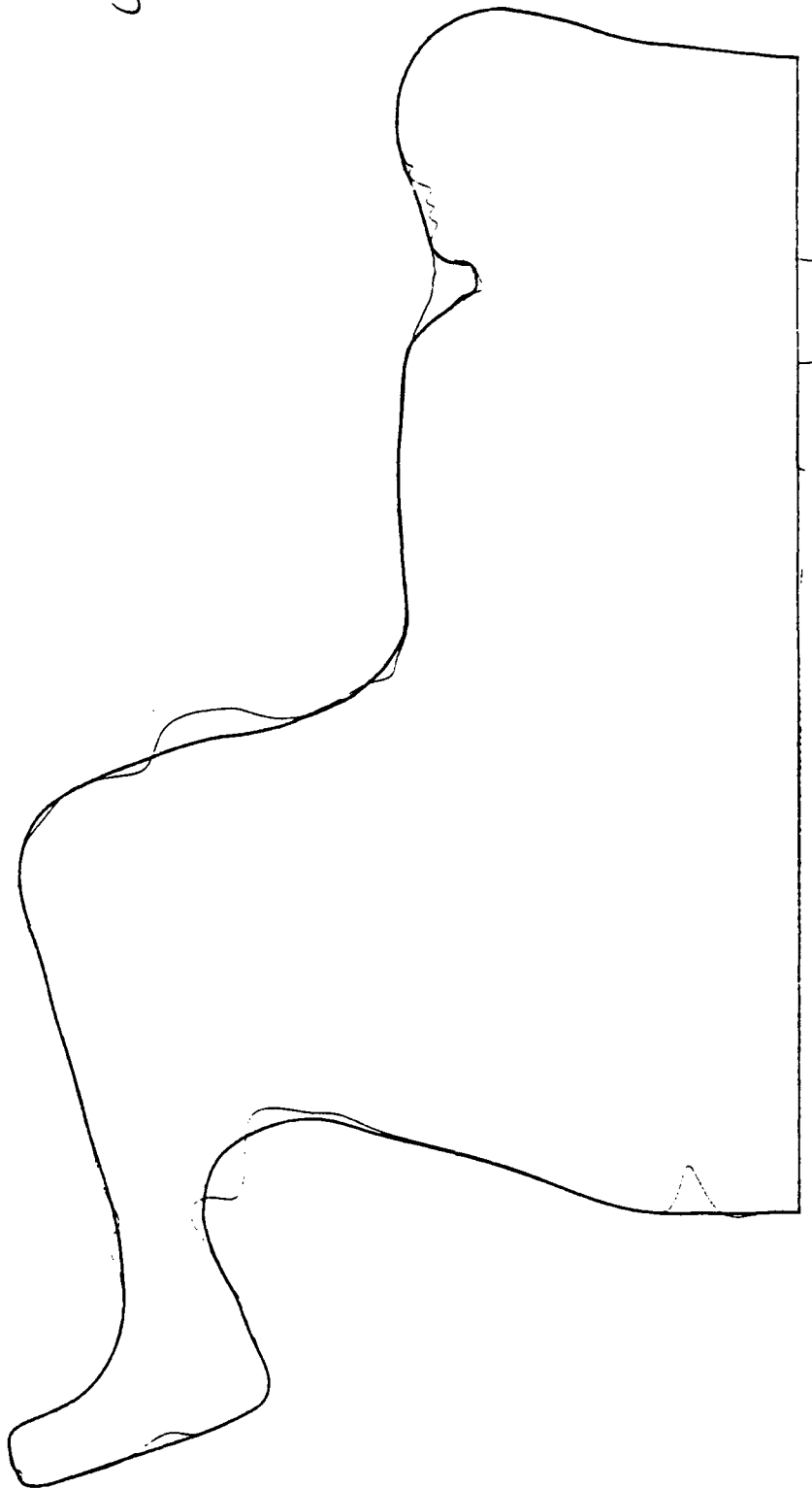


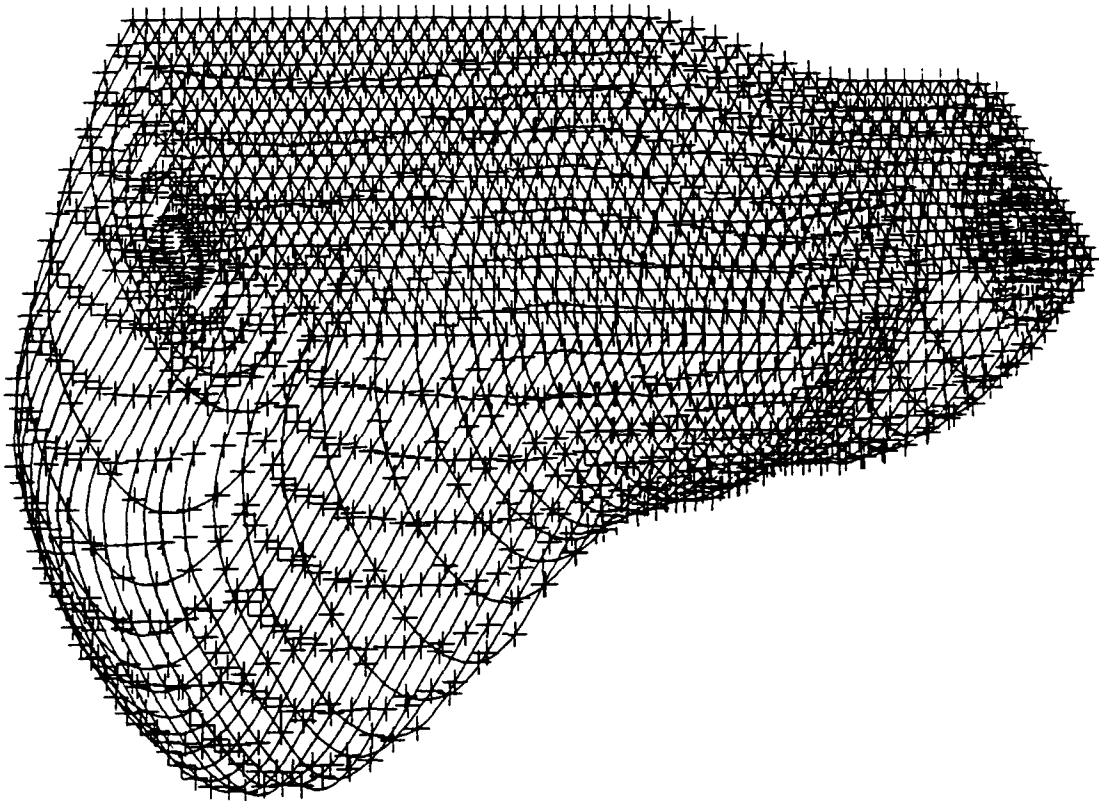
Figure 1 Half Scale F-106 Ejection Seat Wind Tunnel Model

Diagram #2



This is the three digitized drawings overlayed one on top of another on top of another creating one drawing out of three drawings.

# Diagram #3



## REFERENCES

- a. Neal, J. A. (1958). The Development of Methods for Escape from High Speed Aircraft (Report No. 57WC-11100). Wright Patterson Air Force Base, OH: Air Research and Development Command.
- b. Martin-Baker Aircraft Company Limited. (1986). The History and Development of Martin-Baker Escape Systems (Promotional Pamphlet). Higher Denham, Middlesex, England.
- c. Dobbek, R. J. (1976). ACES II Advanced Concept Ejection Seat. (Report No. MDC J4576A). Long Beach, CA: McDonnell Douglas Corporation.

## WPAFB FIBT STRUCTURAL TESTING: HSAP 1991

Christina Ho, engineering aide

These past eight weeks, I have been fortunate enough to work with warm, interesting, talented people. This is my chance to voice my appreciation and regard for them. When a student goes from a theoretical, academically-based environment to a practical application oriented work atmosphere, one's niche is very much changed and the adjustment in priorities now that one plays a different role, can be enormous. However, the common denominator in both of these types of learning experiences are the people -- people who are willing to help and willing to teach, willing to give constructive criticism and rewarding praise, and people who give human context to all the things we do and learn. What makes any task or lesson worthwhile is the fact that in the end, people are helping other people, and that is the motivation to learn, the motivation to teach, and the motivation to work. My gratitude, like my experience here begins with my mentor, and extends to all with whom I have worked. I wish to thank Mr. Pokorski and I only hope that his experience was at least a fraction as rewarding as mine.

During my eight-week employment term I worked on three major projects: the carbon-carbon wing box test, the TMC torque box, and the Radome test. I learned several plotting programs and used them to present data in graph form from the Rams 180 test and the Vortek arc lamp calibration. I also participated in some high-temperature strain gage testing, and acquired valuable technical skills.

For structural testing, static tests are conducted by placing load on the structures in simulation of actual flight conditions and measuring stress and strain for any given conditions. In preparation for the carbon-carbon test, which will run this winter, my participation involved calibration of the load links and adjustment of the signal conditioning within the data acquisition system. I learned that the twenty tension load links and ten compression links serve essentially as load cells in that they measure amount of load using a wheatstone bridge configuration of strain gages, and will be able to physically meet the space constrictions of the test structure. To calibrate these load links, comparison to a standard load cell had to be made. Using these standard load cells and an airpressure load machine, we applied load to the links, ramping up to +35,000 or -30,000 lbs. The data acquisition program, using a given run profile, took data points by recording the millivolt output in strain along the ramp up and back down. Thus, we could define a curve and a computer factor for the load link and with this calibration, use the load links as load cells to measure load on a test structure.

The output readings from the load links, strain gages, and thermocouples feed into five hundred-twelve channels and signal conditioners. I adjusted the potentiometers within the signal conditioning so that a proper excitation voltage was fed into the strain gages, and so that with a dummy quarter-bridge simulating a strain gage reading no strain, the bridge would balance showing zero counts.

On the TMC torque box structure I performed similar adjustments on the signal conditioning system. In addition, I checked and recorded



lead wire resistance in the strain gages using a volt-ohmmeter.

The Radome nosecone structure was a running test, and my participation included operation of the data acquisition program while observing the coordination of the control and data acquisition halves within a test. This project proved an opportunity for me to work on a test that was in a stage of operation rather than preparation, or presentation, as were certain tests whose results I plotted.

For the Rams 180 test, I learned a plotting routine that allowed me to plot certain columns of data against each other. The variables I graphed on a spreadsheet graphing utility included stresses, deflections, motor temperatures, voltages and current versus pitch and time. Through this, I was introduced to yet another facet of structural testing. Whereas much of engineering is concerned with design in preparation to run a test or methods to obtain data results from the run of a test, I was also exposed to the necessity to find ways to communicate the data collected from the test, and the significance of the conclusions that can be drawn from this data. Thus presentation and interpretation of data become vital. With this understanding, I learned several data processing and plotting routines, including a three-dimensional plotting program which I used to graph thermal flux at given plane from the Vortek arc lamp. Using data from the calibration of this high intensity heating lamp, I could create both topographical and isometric triple-axes graphs, showing contour and z-axis respectively, as flux (btu's).

Finally, I was also able to participate in some technical skill work. Much of the theory and physics behind engineering was elucidated when I could actually see how that theory could be made physically and

and applicably viable. It is one thing to understand how a strain gage measures strain by inducing an imbalance in a wheatstone bridge, but it is quite another to apply a strain gage myself and wire it along with bridge completion, and observe the change in output reading as I bend or stretch the bar. Just as different is the understanding of the Seebeck effect from knowing how to splice a thermocouple to a cable and reading temperature from the EMF. Similarly, the difference between the theoretical plan of a test structure on a blueprint can be widely divergent from the actual, logistically feasible set-up that is eventually achieved.

And that, perhaps, was the ultimate lesson learned -- that the gap between theory and application, between vision and articulation, and between the ideal and the real can be wide and deep, and to build the bridge that spans that gulf is an engineering feat that must precede any other engineering feat, and can only be achieved by the cooperation between people.

## REPORT OF A SUMMER

BY ROBERT C. HORROCKS, JR.

An apprentice is an learner. During this summer through the High School Apprenticeship Program, I did a great deal of learning. Through the projects, their applications, and further study, I can conclude that this summer has been, for me, a most interesting one.

Being in a data control group, one main function of the group is to gather, store, and analyze data. This, for the most part, is done with computer systems. One such system that I have assisted with is the Cyber System. This system will be used to control data for many of the test projects in this branch; projects which are related to the National Aero-Space Plane (NASP) and the F-15 Eagle, among others.

Another project of interest is the PCSA V3.0 Network. This system allows the user a wide array of practical applications. There are database, word processor, and spreadsheet programs available. In addition, the network provides a forum for programming DOS and BASIC.

Practical experience is a goal of the HSAP, and in the projects in this group and branch, I have experienced plenty. On RADOME, a nosecone design is being tested for fatigue failure. The NUTCRACKER is used as one of the systems designed to test projects for fatigue failure. Heat and pressure are the main contributors to fatigue failure; therefore,

they are implemented the most in the test projects.

The HSAP has given me an opportunity to explore the "real world" in project testing. It has given me a chance to learn and experience. One goal of mine coming into this summer was to learn basic circuitry. I am pleased to say that I have completed this objective. This practical learning alone will assist me in my college education, and more specifically, in electrical engineering, my field of choice.

Finally, it should be the goal of HSAP to continually upgrade and improve its programs. Please allow me this opportunity to make a few suggestions. Students are often assigned to engineers for their summer. This gives the student little opportunity to learn about the mechanics of the subject they are learning. Many times, students get paired up with computers. Fine. I would want to also be able to physically assemble a test specimen. I am asking for a more "hands on", interactive role with equipment. Second, I would like to see all students given the opportunity to get a security clearance. Almost all facilities have some classified project at the facility, and although it may not be particularly necessary to have clearance, I would feel that it would enhance the experience. Third, I would like to see the student's supervisor, or lab focal point, be required to "rate" the student's progress and overall work capability. This evaluation would assist the student in future work encounters. Fourth, I would request that the program be extended from eight weeks to ten weeks. This is feasible on most students' part, as schools often have an eleven-to-twelve week summer break. Also, this extension would allow

more student interaction with projects, as many projects possess a time frame of more than eight weeks. Finally, I ask that taxes be deducted from the paycheck. This will allow the student to pay less tax come April.

I found this summer to be a good experience. I hope that it will be kept part of the Air Force in the future.

## MY SUMMER AT THE AIRCRAFT SURVIVABILITY RESEARCH FACILITY

Cathie Moore

This past summer I worked in the Aircraft Survivability Research Facility. This is a facility consisting of four gun range to test military aircraft and parts of military aircraft. During my summer I had a few small projects, but most of my time was spent watching and learning from the other engineers. The two main tests I watched were Dry Bay testing and the testing of the flares carried on aircraft to deflect the heat seeking missiles. I used my observations from these tests and some research I did from other tests to compile a brochure for the facility. The other project I had was to calculate the number of gallons of fuel in the scrap fuel tank at each inch increment.

The scrap fuel tank is a total of ninety inches tall and my boss gave me the other dimensions of the tank and my job was to calculate the number of gallons for each inch height. This would enable him to use a dipstick to find the height of the fuel and then he would come back to his office and look at the chart I

made and know how many gallons of fuel were in the tank. I used my knowledge of trigonometry and calculus to figure out the amount of the first inch, then I wrote a program to calculate each inch to ninety inches. This made the work much easier on me because I didn't have to calculate each inch. The computer did it for me.

The brochure I worked on took up a good deal of my time. There was an older version of the brochure and I researched the newer testing and the updated capabilities of each range. I then took my updated version to my boss's boss and he looked at it and made some revisions. After making those corrections, I chose some pictures to go with the text. I then organized the photos and the text and gave it back to my boss's boss. We took it to Tech Photo so that they could make color copies of it.

The Dry Bay testing was done in Range 3. This is the only range at the Base or anywhere else in the free world that is equipped with two jet aircraft engines. These engines blow air across the test article during the test. The test article consisted of a bay in which the electronic wires, avionics and a type of fire extinguishant are contained in front of a fuel bay. A

Soviet round was fired into the article. The test was to see if the fire could be suppressed with the amount and type of fire extinguishant put into the bay would put out the fire.

The testing in Range 2 was testing of the flares carried by almost all of our military aircraft in case of an attack by heat-seeking missiles. The aircraft would drop the ignited flares and hopefully the missile would go after the flares instead of the jet's exhaust. The testing was to see what happened when the flares were shot while inside the aircraft. The cause for concern was that there was a major fuel line right next to the flare dispenser in the F-16 aircraft. There were some pretty spectacular explosions when they went off.



## A SUMMER AT FIBT

Eugene M. Paige  
Project Engineering Aide

During my tour of duty at the Structures Testing Branch of Flight Dynamics Directorate, I gained experience in all of the varied processes that are used. I worked on many of the different projects in Structures Testing: the ETAP (Elevated Temperature Aluminum Project), TMC (Titanium Matrix Composite) and the NASP (National Aerospace Plane) Task D Carbon-Carbon Wing Test. Due to the nature of structural testing, I was not able to see any of these projects go to completion. This is because most of the data and results which come from structures testing are obtained in a very short period of time which comes after a very long period of preparation. However, I hope to come back during my Christmas break from college to see some of the results of these tests.

My summer experience began with my mentor, Mr. Amar Bhungalia. To prepare me for the work which he wished me to do, he gave me some background material to read. One was a history and overview of structures testing in general, and the other a summary of the ETAP program.

There are three main components used in structures testing. First, there is the strain gage. This is a small piece of electrical circuitry that functions by measuring the change in resistance that is caused by the change in shape that occurs as it deforms, i.e. if it expanded when it bent, the resistance would increase, and this change in resistance is used to

measure strain (a dimensionless unit, like microinches per inch, or millimeters per centimeter, etc.). Knowing the stresses that are put upon an object, various things such as stress/strain curves can be calculated.

Second, there is the thermocouple. This is a pair of wires which function like a small thermometer. They electrically measure the temperature at a point very accurately. They are important in tests involving survivability of materials at high temperatures.

The third important component is the load cell. This measures the amount of stress (force per unit area) placed upon an item. The measurements it gives are important in proportional control of stress. What this means is that a certain amount of stress measured will produce a certain voltage (the "output voltage"). This value will be compared by a controller to the voltage of the desired stress (the "input voltage"). The difference in voltages is then put through an amplifier, and the voltage thus produced is used to change the stress. Thus, as it nears the desired stress, the voltage difference decreases and the rate at which it approaches this stress decreases as well. This insures that the desired stress (also known as "load") will be achieved.

The ETAP project, which Mr. Bhungalia wished me to work on, is to create an aluminum compound capable of maintaining its structural integrity at high temperatures. This is useful in airplanes, because aluminum is a very lightweight yet strong metal. The problem is getting it to survive at the high temperatures produced by both the friction from the air and the heat from the plane's engines. The task of formulating and

creating the actual compound is left to the contractor, Lockheed, but testing it to see if it will survive is the job of the engineers at Structures Testing. Mr. Bhungalia's project was to create a test structure which could apply loads to a piece of the ETAP compound, and to that piece alone. If a load was just haphazardly placed upon it, there is a possibility that the test structure, the floor, the wall or almost anything but the desired component would absorb some of the stress. Therefore, it is necessary to design a structure specifically for each project.

Mr. Bhungalia had a blueprint design drawn up for this structure. However, the floor workers who manufacture and assemble the parts for all of the structures can't work from a blueprint - the scale is simply too big to be able to fit all of the information necessary in. Therefore, he wanted me to draw orthographic representations of these parts on the CAD (Computer Assisted Drafting) system that they used. Luckily, I had previous experience in this area, having taken Engineering Graphics 141 over the previous summer at Ohio State University. It took a while for me to get used to the intricacies of their CAD system, but after I learned how to use it, I found it to be a very powerful tool. I had learned the theories of how to draw things in the classroom, but this was good practical experience, which there is no substitute for. An example of these drawings is on page 9.

After I completed the drawings for Mr. Bhungalia, I did drafting for other members of the Project Engineering Group. I drafted Mr. Kenneth Leger's graphite heating system, a patented invention of his which is used for many different tests. This involved more detail than Mr. Bhungalia's

drawings, because it was on a much smaller scale and required numerous notes. I also drafted an expansion support for Mr. Robert L. Schneider. This is a support designed with small gaps so that when it undergoes thermal expansion as a result of being heated, it will not warp or damage the test article. This involved another new experience for me, at least on this particular CAD system: drawing in isometric, or 3-dimensional, views. One of Mr. Schneider's drawings is on page 10.

At this time, I started doing some work for the Instrumentation Group, and moved away from the theoretical side of things and toward the practical side. The experience that I had here gave me appreciation for just how much work goes into putting together the things that the engineers design.

For my first job there, I had to make the one thing required for every test in the building: cables. Cables send instructions from computers to hydraulic actuators, and from load cells back to computers for data gathering and interpretation. These cables can't just be pulled from thin air, though: they must be made in the instrumentation group lab. They must be cut to the length specifically required, and labeled for identification purposes. Then they must be fitted to the job, using different plugs depending on its use.

I worked on many other of the Instrumentation Group's jobs, such as the welding of thermocouples. They must be welded into very specific places along a structure. This is because when an engineer wants to know the temperature at a certain point on a structure, he wants to know what it

is at that point, not one 6, 4 or even 2 inches away. The structure which we were welding onto (the TMC) was directly over our heads, approximately 5 feet off the ground, so we had to reach up and weld the thermocouple wires using electrical welders. Since the wires were round, the tip of the welder had a tendency to slide off to one side or the other, and you would have to attempt to weld the wire again. Also, the weight of the wire was greater than the weld could hold for a great period of time, so you would then have to fasten the body of the wire to the structure by welding it on with pieces of mu metal. There was another hazard which was soon learned from experience: if the welder was used too close to its ground, an arc could jump. This wasn't usually dangerous, but it was definitely a bit disconcerting to have a giant spark jump in front of your face. After the wires were welded to the structure, the free ends had to be attached to a piece designed to hook into a thermocouple reader. It was necessary to test the thermocouples by connecting them to a reader and seeing if the temperature reading went down when the ends were sprayed with circuit cooler. If the temperature reading went up, the terminals were backwards and had to be switched. At first, testing all of the thermocouples seemed to me like wasted time. But it is definitely not. If they aren't properly installed, any results from the test will be useless. So the time to find out if they are faulty is before any test occurs.

It is strange to think about this - many times in our society it is just taken for granted that certain things will be done correctly. But when the business is testing parts of aircraft, parts which will almost surely cause

disaster if they are faulty, nothing can be taken for granted. Everything must be checked and double checked to make sure the checker didn't make a mistake. This shows up in other things done in Structures Testing. In some cases, there are two computers attached to the same test getting independent readings to make sure that if something is wrong with one, the other will catch it. Again, it just reflects the fact that what is being done has to be very exact.

After this experience with the thermocouples, I learned how to lay strain gages. Laying strain gages requires a very delicate touch, as it involves paper-thin circuitry which can be damaged by being deformed. First you must clean the surface it is to be placed upon by sandblasting and rubbing with both acidic and alkaline cleansers. Then the gage is put where it is to rest, it is lifted and catalyst for curing of the adhesive used is put on the bottom. Then the glue is put onto the specimen, and the gage quickly attached. After a short time, you test the gage to be sure it is secure. Then you must solder wires onto the solder tabs so that the resistance change may be measured.

After my experience the the strain gages, I went back to the CAD system to do another drawing for Mr. Leger. This was my greatest challenge, and my accomplishment of which I am the most proud. He needed to submit a report upon the Carbon-Carbon Wing Test project, which required a picture of the finished product. Unfortunately, the project wasn't completed yet. So, Mr. Leger gave me an orthographic front view of the test structure that he had drawn, and asked me to make an isometric drawing of

it. This was a challenge, because there are many layers to this structure, and each layer covers up another as you go along. Thus, I had to constantly clip and trim my work to keep it true to the isometric view. Also, Mr. Leger desired that it should be fairly true to detail, so if it was zoomed in to look at a specific part, that part would appear as it would in real life. Thus, I did most of the drawing one piece at a time, maintaining great detail on a very small scale. The end result is included as page 11 of this report.

After doing this, I worked on the Carbon-Carbon control panel for my last project. Labeling wires is quite time consuming. Labeling the control panels they go into and installing them is more so. In many cases, counting the various necessities such as thermocouple wires, strain gage cables, load cell cables and control cables, the number of channels used for a project can easily exceed five hundred. It is necessary to label all parts of the control panel - the front so that you can see where to plug them into the computer from, the back so you can see where the cables go to when they come in. Also, in many cases the channel number on the computer is different than the thermocouple, load cell or strain gage number. Thus, two numbers must be used to describe them, and two labels made. After the labeling is done, installation begins. The cables are generally wrapped in bundles of over one hundred, and it takes a great deal of time to sort them and straighten them out so that they may be installed. The feeling of satisfaction that you get from seeing a completed control panel is quite worth the effort, though.

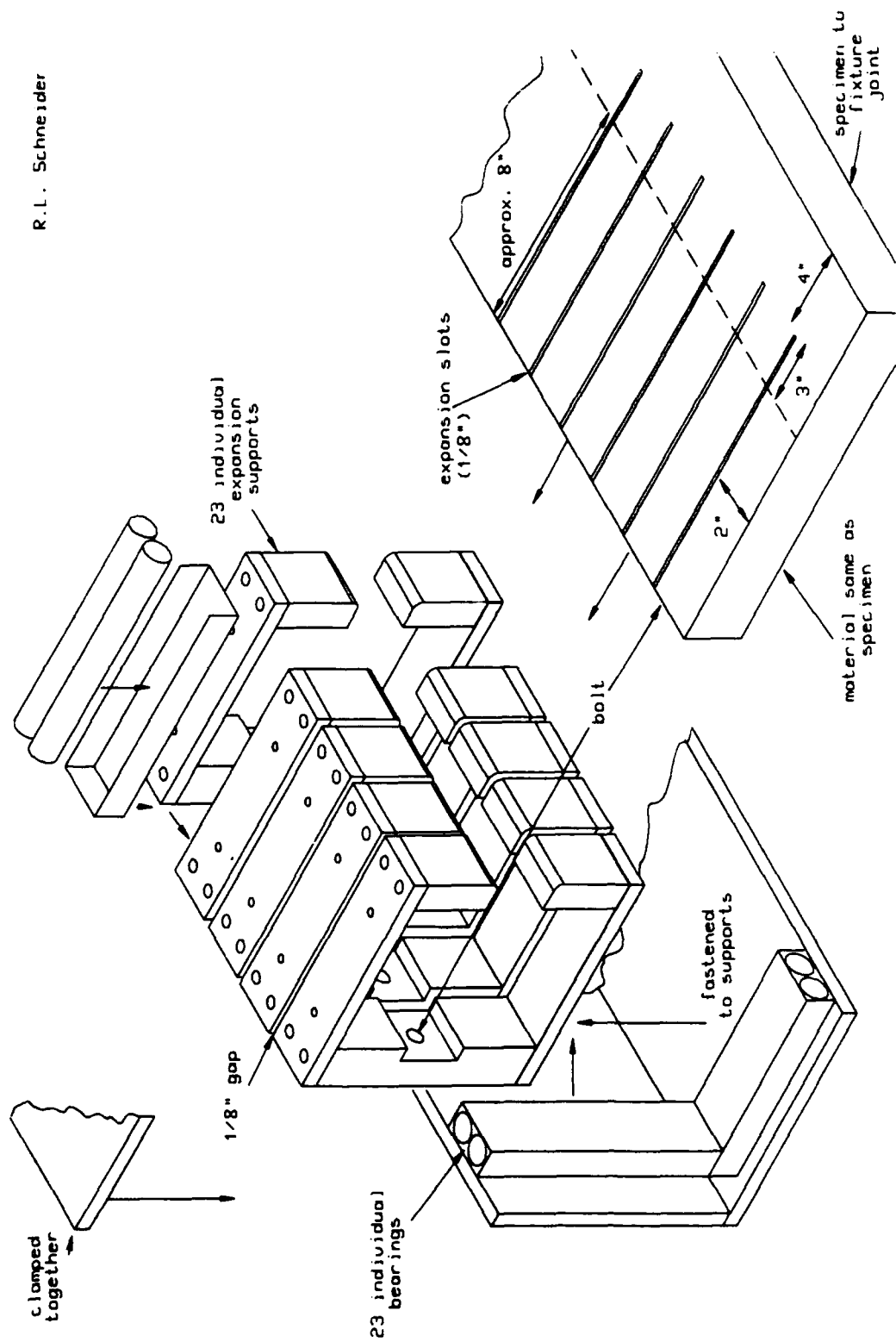
All in all I regard my summer at FIBT as a useful learning experience. I learned a great deal about stress and strain. It is hard to imagine the

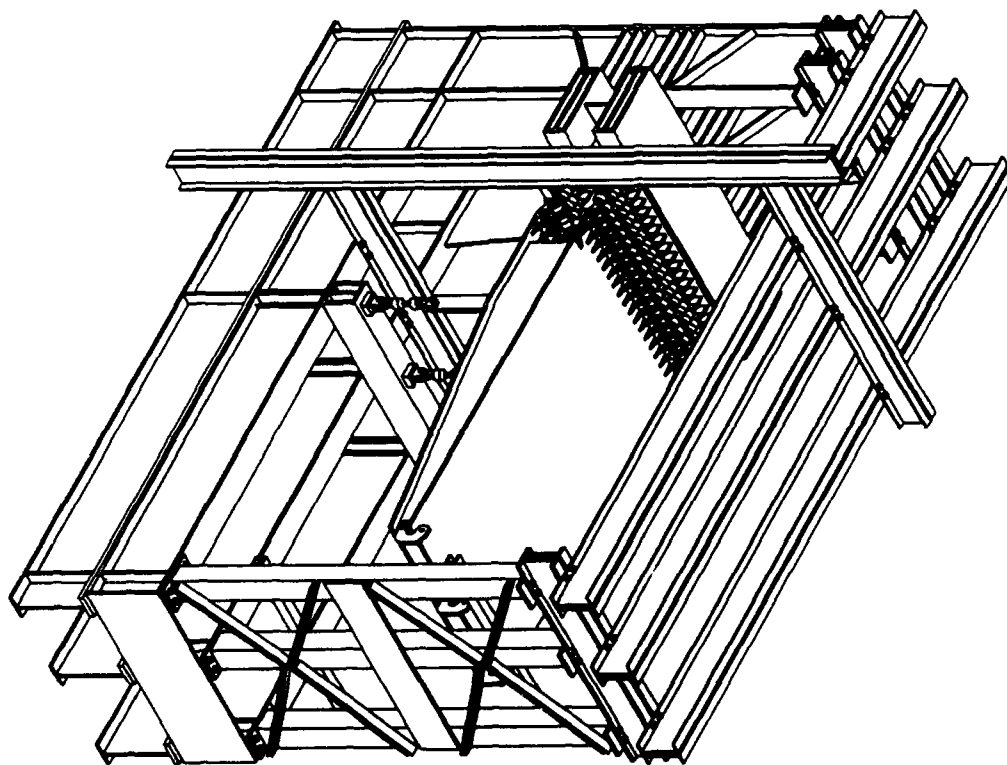
something which represents a small part of a small chapter in my physics book can be a life's work for some people. Although I have held a part-time job for a year and a half, the experience of having to work full-time is definitely a different one, and one which should prepare me well for a career later in life. I would definitely recommend the experience to anyone who has doubts about it.





R.L. Schneider





## HIGH SCHOOL APPRENTICESHIP PROGRAM FINAL REPORT

Michael I. Richie

During my summer apprenticeship, my main job was organizing the group's database. Their database contained reports, conference proceedings, proposals, and symposiums. My job was to organize this information and then place it into a computer database for easy retrieval.

### INTRODUCTION

I was placed in the Structural Division at Wright-Patterson Air Force Base in Dayton, Ohio. I was assigned to the Vibration Group, which performs vibration tests on structures to be used for aircraft. Although I did not participate in any of these tests, the work I was given benefitted the group and enriched my computer skills.

### PROBLEM PRESENTATION AND RESULTS

Although the database was my main job, I also did view-graphs, charts, and computerized hand-outs for the group. I enjoy computer work and all of this enabled me to do what I liked. The work also introduced me to new programs which enable one to enhance his projects. Such programs, like Harvard Graphics, GEM, and Corel Draw, I had heard of, but never had been able to use.

When I came, the main work for me had already been planned. I would first organize the database and then enter in the reports, symposiums, etc. The group had already attempted to have a database but it really did not get anywhere. When I first started the database, I used Enable. Soon, that was decided that it was not as User-friendly as the Microsoft database so I switched all the reports from the Enable Database to the Microsoft Database. However, in the Microsoft Database, there are the reports from the vibration group plus reports from other groups in the building.

The vibration group still has a database in Enable, but it can also go to the Microsoft to look for its needed reports.

#### CONCLUSION

This is what my summer apprenticeship basically consisted of. The first two weeks alone were spent trying to modify the Enable database and then most of the rest of the time I was filling it up with the reports. I would not say that this was a research apprenticeship, but more of an enrichment one. I enjoyed working here and I enjoyed the work that I did. It was fun and I probably will apply for the program again next year.